UNIVERSITY OF GENOVA

PHD PROGRAM IN BIOENGINEERING AND ROBOTICS

On the Use of Large Area Tactile Feedback for Contact Data Processing and Robot Control

by

Alessandro Albini

Thesis submitted for the degree of *Doctor of Philosophy* (32° cycle)

May 2020

Giorgio Cannata — Supervisor

Giorgio Cannata — Head of the PhD program

Dibris

Department of Informatics, Bioengineering, Robotics and Systems Engineering

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Alessandro Albini
May 2020

Abstract

The progress in microelectronics and embedded systems has recently enabled the realization of devices for robots functionally similar to the human skin, providing large area tactile feedback over the whole robot body. The availability of such kind of systems, commonly referred to as *robot skins*, makes possible to measure the contact pressure distribution applied on the robot body over an arbitrary area. Large area tactile systems open new scenarios on contact processing, both for control and cognitive level processing, enabling the interpretation of physical contacts. The contents proposed in this thesis address these topics by proposing techniques exploiting large area tactile feedback for: (i) contact data processing and classification; (ii) robot control.

List of Publications

Part of the contents proposed in this thesis has been published in the following journals or conferences.

Journals

- The International Journal of Robotics Research (2020) - A. Albini and G. Cannata - *Pressure Distribution Classification and Segmentation of Human Hands in Contact with the Robot Body*
- MDPI Sensors (2019) - W. Wasko, A. Albini, P. Maiolino, F. Mastrogiovanni and G. Cannata - *Contact Modelling and Tactile Data Processing for Robot Skin*

Conferences

- RO-MAN 2018 - A. Albini and G. Cannata - *Tactile Image Generation from Contacts Involving Adjacent Robot Links* - (KROS Best Paper Award Winner)
- HUMANOIDS 2017 - A. Albini, S. Denei and G. Cannata - *Enabling Natural Human-Robot Physical Interaction using a Robotic Skin Feeedback and a Prioritized Tasks Robot Control Architecture*
- RO-MAN 2017 - A. Albini, S. Denei and G. Cannata - *On the Recognition of Human Hand Touch from Robotic Skin Pressure Measurements Using Convolutional Neural Networks* - (Best Paper Award Winner)
- IROS 2017 - A. Albini, S. Denei and G. Cannata - *Human Hand Recogniton from Robotic Skin Measurements in Human Robot Physical Interactions*
- IROS 2017 - A. Albini, S. Denei and G. Cannata - *Towards Autonomous Robotic Skin Spatial Calibration: a Framework Based on Vision and Self-touch*

Table of contents

List of figures

List of tables

Part I

Preamble

Chapter 1

Overview

This Chapter provides an overview of the document. Section 1.1 motivates the use of large area tactile sensors in robotics. The objectives of the thesis are then introduced in Section 1.2. Finally, the structure of the manuscript is described in Section 1.3.

1.1 Skin for Robots

Until a few years ago, robots have been used to replace humans in repetitive tasks requiring high accuracy and they have been kept in separate workspaces for safety reasons. In the future, robots are expected to closely cooperate with humans in unstructured environments, supporting them in a variety of activities, and to accomplish complex tasks otherwise difficult to be tackled, or tedious and error prone. To this aim, and in order to ensure safe interactions, robots are expected to embed human-like sensing modalities such as vision, touch, speech, allowing humans to interact with them in the most natural way.

In the literature, human-robot interaction (HRI) has been largely based on vision systems, for example to recognize gestures (Li, 2012), to cooperate with robots in assembly tasks (Kimura et al., 1999) and to deal with collision detection problems (Ebert and Henrich, 2002). When contacts occur or a continuous interaction with the robot is required, the capability of sensing the contact phenomena is fundamental. However, when physical human-robot interaction (p-HRI) is considered, problems of safety arise. To this aim, force/torque sensors have been largely used in order to ensure a safe p-HRI, by detecting collisions (Haddadin et al., 2008) and ensuring robot compliant behaviours in response to external forces (Duchaine and Gosselin, 2007; Grunwald et al., 2003).

Humans perceive contacts mostly through the skin; therefore, tactile sensors mimicking its functionality and integrated on the robot body are expected to provide additional information with respect to force/torque sensors. The progress in microelectronics and embedded systems

has recently enabled the realization of devices for robots functionally similar to the human skin, namely *robot skin*, which have been proposed by several authors (Cannata et al., 2008; Cheung and Lumelsky, 1989; Minato et al., 2007; Mittendorfer and Cheng, 2011; Mizuuchi et al., 2006; Mukai et al., 2008; Ohmura et al., 2006; Someya et al., 2004; Um et al., 1998). The robot skin is a large-area tactile system (possibly) composed of thousands of sensors (i.e. **taxels**) distributed over the robot body, which can sense pressure, proximity and temperature just to name a few, allowing to describe contact events from different points of view.

Although robot skin can be used as a safety device, for example to detect a collision and its intensity, it would be an understatement to restrict its domain of application only to safety. Indeed, such a kind of distributed system allows to capture complex contact phenomena occurring on the whole robot body. Bicchi et al. (1993) have shown that for a given robot geometry for contacts over small areas it is possible to reconstruct the interaction forces and the contact centroid location by processing lumped force/torque measurements. Although this method has been proven effective for object manipulation using robot hands, it can be hardly scaled in case of multiple contacts, or complex interactions expressed over large areas, which are phenomena expected to arise in tasks involving tight HRI or when large areas of the robot body are in contact with the environment. On the contrary, the robot skin should make possible to measure the contact pressure distribution applied on the robot body over an arbitrary area. Thus, beyond safety aspects, the robot skin opens new scenarios on contact processing, both for control and cognitive level processing, enabling the interpretation of physical contacts.

1.2 Objectives

As discussed in the previous Section, large area tactile sensors can be used to improve the capabilities of robots working with humans in unstructured environments, where unpredictable contacts can occur. In such a scenario it would become of interest to understand the origin of the contact (e.g. if generated from contacts with humans or the environment) or its purpose (a human performing a push, pull, etc.), thus enabling appropriate robot reactions. For example, if a collision with the environment is detected, the robot can react trying to avoid the obstacle or to guarantee non-destructive motions. On the contrary, if a contact with a human is detected, the robot can behave differently trying to accommodate the human intentions. In order to realize this kind of robot behaviour, the overall problem can be split in two levels: (i) to process and classify tactile data in order to interpret the contact event; (ii) to command robot motions based on the intensity of the contact and its location on the robot

surface.
The goal of this thesis is to make a step in this direction, proposing methods taking advantage of robot skin feedback to process large area contact distributions and to control the robot motions. In particular, the following points are addressed:

- how contact information can be encoded using a standard representation which is easily portable to different robots or tactile sensing technologies;
- the processing and classification of robot skin measurements, to extract and interpret contact information;
- the usage of large area tactile feedback to control robots operating in unstructured environments or with humans.

1.3 Outline

This document is structured as follows. In Chapter 2 a review of the literature is presented. Chapter 3 describes a sequence of elaboration steps allowing to encode robot skin measurements into tactile images. Chapter 4 addresses the problem of using robot skin to recognize contacts generated by a human touch. This topic of tactile data classification is further investigated in Chapter 5 by testing the performance of the system with respect to variations on the input. Chapter 6 resumes the contents introduced in Chapter 3, discussing how to extend their applicability to the whole robot body. In Chapter 7 the problem of reconstructing the spatial arrangement of the robot skin tactile elements is addressed. Chapter 8 proposes a technique exploiting large area tactile feedback to control the interaction between the robot and the environment. Chapter 9 is focused on the p-HRI scenario and presents a preliminary attempt to join the contents discussed in Chapters 3 and 4 with robot control aspects. Conclusion follows.

Chapter 2

State of the Art

The development of tactile sensors has begun in the 1980s (Dahiya et al., 2013). The first prototypes were initially thought to support robots in manipulation tasks and thus designed to be applied to custom robot parts such as hands or fingertips. During the years, the technology rapidly evolved allowing to design devices that can be integrated on different parts of the robot body, covering *small* areas.

So far, in the literature, a wide variety of tactile sensors has been presented. They mainly differ from each other in terms of transduction principles, materials, conformability, spatial resolution, size, power consumption and robustness (Dahiya et al., 2013). Due to such amount of available options it is still quite hard to find a solution that prevails with respect to others. As a consequence, unlike cameras, there is not an unique technology for tactile sensing yet, and the choice of one with respect to another usually depends on the specific application. Moreover, tactile sensors can provide different types of information, e.g. normal forces acting within the contact area, temperature, shear forces, lumped forces, etc. Clearly, this large amount of options does not allow to define standards from the software point of view either: algorithms and data structure are usually tailored for a specific sensing technology.

New applications, mainly related to HRI and motion planning in unstructured environment, increased the interest of the research community in developing tactile sensors covering *large* areas or possibly the entire robot body (Dahiya et al., 2013). This transition from small to large area is not simple, therefore the problem of tactile sensing must be tackled from a different point of view, considering the whole system rather than the single sensor. When the number of sensors scales from tens to thousands, aspects related to wiring complexity, overall system weight, networking solutions, power consumption and realtime data processing, must be taken into account.

Instead of proposing an exhaustive review of the literature related to tactile sensing, this Chapter presents an overview on the current state of the art focusing on three aspects which

are related to the topics addressed in this thesis. A selected set of results is presented in order to clearly frame the contents discussed in this document with respect to the literature. In particular, Sections 2.1 and 2.2 discuss the use of tactile sensors, both for the small and large area in applications related to contact processing, classification, p-HRI and robot control. Section 2.3 focus on large area tactile systems, describing the issues and the technique proposed in the literature to build robot skin data representations.

2.1 Tactile Sensors for Contact Processing and Classification

Applications requiring to process or classify the *contact shape* need to retrieve it from the raw output of the tactile sensing system. Humans are able to reconstruct contact shapes by processing stress and strain information originated from mechanoreceptors located in the skin (Serino and Haggard, 2010). In robotics, stress and strain measurements can be used to reconstruct the contact shape and exerted forces exploiting contact mechanics principles (Johnson, 1985). The problem has already been addressed in the literature considering large area tactile systems (Muscari et al., 2013; Seminara et al., 2015), focusing also on computational aspects (Wasko et al., 2019). However, model-based techniques have some disadvantages. Firstly, they strictly depend on the transduction technology. Secondly, due to the complexity of the contact mechanics, contact models are based on quite restrictive assumptions, valid considering small or planar surfaces, but that can hardly scale, making difficult to be applied to generic types of robot skins.

Due to the aforementioned issues, an *approximate* reconstruction of the pressure distribution is usually preferred. The most popular approach consists of converting the pressure data distribution into a *tactile image*, which is a representation where the intensity of each pixel is computed from the raw sensor value. Although tactile images do not exactly represent the real contact shape, they allow to easily process or classify contact information using state of the art image processing techniques.

The majority of the works proposed in the literature exploits tactile images in applications requiring to process and classify the contact shape. Schneider et al. (2009) used a small pressure array integrated onto a robot fingertip to actively touch objects of interest and the resulting tactile images were classified using a Bag of Visual Words model. Liu et al. (2012) covered a robot hand with small planar tactile patches mapping the whole pressure readings onto a single image. Finally, they trained a neural network to classify a set of grasped objects. Cao et al. (2016) used a stream of tactile images obtained during a grasping task to classify

10 different objects using a convolutional neural network. Gandarias et al. (2018) proposed an approach where a high-resolution patch of pressure sensors integrated on a gripper is used to classify the tactile images generated by objects, human limbs, and fingers through a convolutional neural network.

Besides the use of robot hands, other approaches employ a rectangular patch of tactile sensors mounted on the robot end-effector. Pezzementi et al. (2011) proposed to obtain a set of tactile images generated from a sequence of contacts and used a Bag of Visual Words model for object recognition. A similar approach has been considered by Luo et al. (2015b) in order to classify a set of objects using an innovative *tactile SIFT* descriptor (a specialization of the *SIFT* algorithm originally developed for image data processing). The extracted features are then classified using the Visual Bag of Words algorithm producing good classification results. Taking advantage of the similarity between tactile and visual images, the same authors proposed algorithms to merge tactile and visual feedback for object localization and classification (Liu et al., 2017; Luo et al., 2015a).

2.2 Tactile Sensors in Physical Human-Robot Interaction and Robot Control

Tactile sensors in physical human-robot interaction (p-HRI) have been mainly exploited as an input for two reasons: (i) to implement touch-based robot control strategies; (ii) to recognize touch-based human interactions.

Focusing on p-HRI domain, tactile sensors have been widely employed to implement touch reactive control laws. Wosch and Feiten (2002) showed that patches of pressure sensors integrated on a robot link allow human operators to guide a robot arm. The pressure readings are translated into motion vectors used for controlling the arm position. With the same goal, Cirillo et al. (2016) mounted a tactile patch on a KUKA robot forearm, allowing a user to manually move the robot through multiple points. Furthermore, they specified a safe robot behavior depending on the magnitude of the force applied by the human operator.
Similarly, Schmidt et al. (2006) used an array of capacitive-based pressure sensors mounted on a robot gripper to implement a control strategy allowing the robot to adapt its posture in response to the force applied by a human operator.
Frigola et al. (2006) implemented a compliant behaviour in a robot arm exploiting the feedback of a force sensitive bumper skin. Leboutet et al. (2016) achieved a whole robot body compliance by using a technique based on hierarchical force propagation exploiting force feedback provided by an artificial skin. Differently from the model-based approaches

discussed above, Pugach et al. (2016) presented an admittance controller based on a self-organizing map, which is trained by associating the position of the contacts (retrieved using tactile sensor feedback) with the torque measured from the joints. With a different goal, Nguyen et al. (2018b) presented a framework allowing humanoid robots to move in a safe zone when interacting with humans. In this work, tactile sensors are used to build a peripersonal space around the robot. The peripersonal space proposed by the authors has also been exploited in Nguyen et al. (2018a). The paper addresses the problem of integrating p-HRI and s-HRI (social HRI) in a object handover task.

Tactile sensors have also been implemented to recognize different *touch modalities*, namely actions (e.g. *Pat*, *Push*, etc.) performed by human subjects using the hand. The general approach is similar in most of the techniques proposed in the literature: a set of features is extracted and classified using supervised machine learning algorithms (e.g. Silvera-Tawil et al. (2015)), the main differences among the various solutions being the number of modalities classified and the training methodologies adopted. In particular, Naya et al. (1999) used a *k*-neighbor algorithm to classify 5 touch modalities, based on data collected in experiments involving 11 users. A neural network has been considered by Stiehl and Breazeal (2005) in order to classify a set of 8 interactions performed by a single subject. Similarly, Ji et al. (2011) classified 4 touch types using the SVM algorithm. Tawil et al. (2012) used the *LoogitBoost* algorithm (Friedman et al., 1998) to recognize 9 touch modalities acquired from 40 subjects. Finally, Kaboli et al. (2015) implemented an SVM to recognize 9 touch modalities using a multimodal robot skin providing pressure, acceleration and proximity measurements.

Beyond p-HRI, tactile-based control has been used to explore objects or the surrounding environment, usually to detect edges or reconstruct the shape of objects. Early works (Berger and Khosia, 1988; Muthukrishnan et al., 1987; Ning Chen et al., 1995) take advantage of tactile images in order to detect edges in real-time. Image processing techniques have also been used by Berger and Khosla (1991). The authors made a step further using tactile sensors in the feedback loop of a manipulator to track edges in real time. The same problem has also been addressed by Chen et al. (1995), proposing a tactile servo scheme. Li et al. (2013) presented a control framework for tactile servoing. Contact features are extracted from images and the robot is controlled to realize different tasks, such as force control and object exploration.

Other approaches proposed control strategies where the object is actively touched (usually with a sensorized fingertip), and the tactile feedback is used to adjust the position of the end-effector in order to enhance the perception of features of interest (e.g. edges). Martinez-

Hernandez et al. (2013) proposed a strategy where the robot collects tactile information by tapping against the object. This strategy is used to perceive edges orientation and therefore optimize the position of the fingertip during the exploration. A similar technique has been applied by Lepora et al. (2017) in a task of object contour following. Tactile feedback is used to regulate the position of the fingertip aiding the perception of edges and ridges.

Touch based closed loop control laws have also been proved to be effective in fine manipulation tasks. Using a sensorized fingertip mounted on a robot end-effector, Cramphorn et al. (2016) showed the possibility of rolling a cylinder and control its position. With a similar goal, Tian et al. (2019) proposed a framework combining deep learning and model predictive control to achieve fine manipulation of three different objects. A fingertip is mounted on a 5-axis machine performing three tasks: (i) ball repositioning; (ii) joystick deflection; (iii) die rolling to reach a specified face.

Exploration and manipulation tasks have also been experimented using large area tactile systems. An example can be found in Calandra et al. (2015). Considering a scenario where the robot arm is in contact with an object, the skin measurements are used to train a model that computes the torques needed to compensate the contact. The work proposed by Jain et al. (2013) addresses the problem of moving a robot arm in a cluttered environment. Authors exploited a control architecture based on a quasi-static Model Predictive Control (MPC) on top of an impedance control, enabling robot compliance when in contact with obstacles. The one step horizon MPC is used to set constraints on the maximum force variation between two consecutive time steps. An additional high level planning strategy is then used to get out of possible local minima. Authors further developed the method (Killpack et al., 2016) and they also added the possibility of using tactile information to classify some properties of the objects (e.g. hardness) and to perform object recognition (Bhattacharjee et al., 2012).

2.3 Large Area Tactile Systems Calibration and Data Representations

A robot skin is a complex system (possibly) composed of a high number of transducers providing information about contact events occurring on the whole robot body. The output of a large area tactile system consists of thousands of raw measurements that should serve a number of applications (or tasks) that can be quite different among each other (e.g. robot control, contact classification, etc.). Thus, the following scenario can be imagined: on one side there is the robot skin which provides raw measurements, while on the other there is a set of applications exploiting tactile feedback. What is missing in the middle is a sort of

representation, working as an *interface* able to provide organized data to higher levels that must respond to contact events. In principle, the raw measurements should be collected and organized, representing them in order to provide a meaningful information for a specific operating context.

Although the output of a tactile system in some cases can be used as it is (e.g. detecting the mean pressure applied), raw measurements alone are usually of low interest. An intuitive way for representing robot skin data is to provide geometric information (i.e. position and orientation of the corresponding sensors on the robot body) along with the sensors measurements. Thus, if the robot skin is *spatially calibrated*, high level applications can exploit geometric information replicating the real sensors placement *as close as possible*. A spatially calibrated skin provides an approximation of the robot body shape allowing to associate tactile stimuli to positions on the robot body.

As close as possible means that sensors position and orientation are generally approximated. This might sound strange, since, in principle, the pose of the taxels could be determined when designing the robot. However, to date, the design and integration of a robotic skin on a robotic platform have always been an a posteriori procedure where the skin has to be adapted to already existing robots (Schmitz et al., 2011). This approach is due to the fact that realizing a complete skin for a robot is a complex and expensive task, thus it is necessary to devise a technology that can be ported to different robotic platforms allowing a large-scale production of the hardware. The first step of adaptation and integration of a robotic skin can be performed with a CAD software. However, the correct position of the single taxels is affected by uncertainty. Flexible robot skins are usually manufactured as planar sheets of sensors (see Figure A.2), that have to be adapted to 3D surfaces. Even with CAD modelling, it is hard to exactly predict how elements belonging to a 2D flexible geometry will be redistributed on an arbitrary surface. When positioning large area tactile sensors on the robot body, problems of adaptability to the surface arise. In general, the robot body can have different curvature radiuses, implying that, even with an a priori estimation of the 3D position of the tactile elements, the integration procedure introduces high degrees of uncertainty, placing sensors in different positions with respect to the desired ones. Furthermore, the integration process is performed by hand and thus error prone by definition. Due to the aforementioned uncertainties, even two robot covers with the same identical geometry can be sensorized in a slightly different way.

It emerged from the previous discussion that constructing an accurate representation of the sensor placement on the robot body is not a trivial task. For this reason, autonomous or semi-autonomous procedures to achieve an accurate spatial calibration have been investigated in the literature. The first example of autonomous skin calibration can be found in Cannata

et al. (2010). The authors implemented a method for discovering the taxels location using a complete knowledge of the robot shape and using an external support for exciting the skin with a continuous and moving touch. The sensors distance from the current stimuli is calculated using a contact model and the position is estimated by triangulating the distances obtained while moving the contact area. However, the approach cannot be performed autonomously by the robot without a properly designed calibration platform. In Del Prete et al. (2011), the sensors arrangement is obtained through measurements of the force applied to the skin using a 6-axis force/torque sensor placed on the shoulder of the robot. The measurements are used to compute force axes passing through contact centroids activating the sensors. The sensor position is estimated by correlating the axes of the forces activating it. The spatial calibration result is then improved using the knowledge of the robot body model. The presented approach strictly requires a force/torque sensor preceding the part under calibration, thus it cannot be easily extended to other parts of the robot body outside of the arms. Another approach is presented in Mittendorfer et al. (2014) using a robotic skin made of hexagonal modules. The sensors location is obtained using the measurements of on-board accelerometers and with a complete knowledge of the shape and topology of the modules. However, this method strictly applies to skin systems with distributed accelerometers and precise and well defined geometrical constraints.

As previously discussed, robot skin can serve a lot of applications with different goals. Although spatial calibration could be fundamental to process contact events, a representation based on raw measurements along with 3D position information could not fit or could not be suitable for some tasks. Therefore, other methods for representing robot skin data have been proposed in the literature. Differently from the spatial calibration, these representations, often referred to as *artificial somatosensory systems*, differ from the real sensors placement, abstracting the robot body shape. Some of these representations are built on top of a spatially calibrated robot skin (Denei et al., 2015; Stiehl et al., 2004), while others are created from scratch (Brock et al., 2009; Hoffmann et al., 2018; Kuniyoshi et al., 2004; Pugach et al., 2015).

Stiehl et al. (2004) proposed a framework where contact events are translated in a semantic alphabet. Using a sensorized robot hand as a platform, some features representing motion, direction, and orientation of a stimulus are computed from sensors position and raw values. Thus, the semantic alphabet is constructed by encoding these features. Although the method provides a high level representation of contact events, topographic information are not considered. Instead, Kuniyoshi et al. (2004) addressed the problem of learning topographic body maps through sensory-motor interaction processes. During the motions, contact events occurring on a simulated agent are temporally correlated to build a somatosensory map of the

body, thus obtaining a topographic representation of temporally correlated locations. A map that also preserves geometric properties can be built assuming that close tactile elements are stimulated together as the effect of the same contact event. However, if more than one contact simultaneously occurs on different (and far) body locations, contacts will be represented close in the map while the real contact locations are far from each other. A similar approach has been considered by Brock et al. (2009), where groups of tactile sensors are activated and processed, (mostly) manually interacting with the robot, in order to create a 2D topographic map by defining connections between correlated tactile elements. Pugach et al. (2015) trained a self-organizing map (SOM) to reconstruct the shape of a sensorized tactile sheet. In the work proposed by Hoffmann et al. (2018), authors create an artificial somatosensory map of the iCub robot body, by semantically clustering the robot areas (torso, arm, finger, etc.). The method is based on a SOM composed of neurons that group the responses from different sensors. The weights of the SOM are learned by stimulating the robot skin.

A different approach has been presented by Denei et al. (2015). Instead of learning the body representation from scratch, the method assumes to start from the 3D spatial calibration of the sensors. A tactile map is then defined by mapping the 3D sensors position on a 2D surface by preserving the geometric properties of the robot skin. Furthermore, the approach allows to obtain a one-to-one correspondence among taxels in 3D and 2D, allowing to easily swap between the two representations when it is needed. Authors proved that tactile maps generated using this approach are suitable for tactile-based control tasks.

Part II

Contact Data Processing and Representation

Chapter 3

Robot Skin and Tactile Images

Tactile images have been proven to be powerful tools when the contact shape requires to be processed. However, since they are structurally different from robot skins, they cannot be directly generated when large area tactile systems are considered.
In this Chapter, the problem of creating tactile images from contacts distributed on the robot body is addressed. The proposed technique makes possible to create a picture of the contact with minimal distortion with respect to the original 3D shape. The possibility of obtaining a 2D representation abstracted from the robot geometry allows to use classical image processing/classification techniques to process/classify contact distributions occurring on complex 3D surfaces.

3.1 Introduction and Motivation

As previously described, when the contact shape needs to be processed (e.g. edge detection, contact classification, etc.) it is common to rely on tactile images. The advantage is that techniques already developed for images can be used for tactile data processing.

It appears clear from the previous discussions (see Sections 2.1 and 2.2) that so far in the literature tactile images have been generated from planar patches containing (in most of the cases) sensors distributed on a regular grid with uniform spatial resolution and generally covering a small area.
When large area tactile systems are considered, there is no such concept of tactile image. Taxels composing the robot skin are placed on a complex manifold with non regular spatial distribution, while pixels in images are distributed on a regular bi-dimensional grid.
By recalling the concepts discussed in Section 2.3, robot skins need abstract representations in order to interface the hardware with high level applications. Having this picture in mind, it can be assumed that some of these applications consist in state of the art techniques already

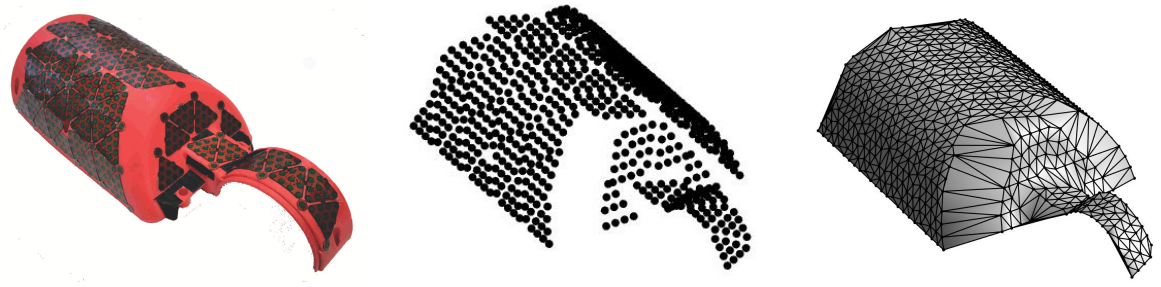
proven to be working with tactile images (e.g. tactile images classifiers, tactile image-based contour following, etc.). An internal representation, capable to represent contacts occurring on the skin as tactile images, is necessary to interface the robot skin measurements with these applications.

In this Chapter, a technique to transform robot skin measurements into tactile images is presented. It will be addressed how to generate images from one single link. The case considering the whole robot kinematic chain introduces additional complexities that will be explained and preliminarily addressed in Chapter 6.
The proposed technique consists of creating a planar representation of the robot body, a *tactile map*, by performing a set of geometric transformations. Contact pressure information can be mapped on the tactile map, thus generating an image that best preserves the contact shape.

3.2 Map the Robot Body onto a Flat Representation

It is assumed to have a robot link covered with robot skin, composed of N distributed pressure transducers (see Figure 3.1(a) as an example).
The position and the response of each taxel to a given pressure stimulus on the robot body are assumed to be known, possibly as the outcome of a calibration procedure. Then it is possibile to define the set $T = \{\mathbf{t}_1, \dots \mathbf{t}_N\}$, where the element $\mathbf{t}_i \in \mathbb{R}^3$ represents the 3D position of the i-th taxel; the set T can be intended as a sort of *point cloud* where each taxel position $\mathbf{t}_i$ is referred with respect to the reference frame of the sensorized robot link (see Figure 3.1(b)).

A *Delaunay triangulation* (Fortune, 1997) applied to T allows to define a list of topological relations F between adjacent taxels, thus creating a 3D mesh $S^* = (T, F)$ representing a

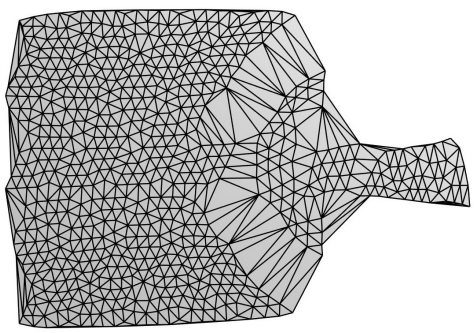


(a) Real robot link covered with robot skin.

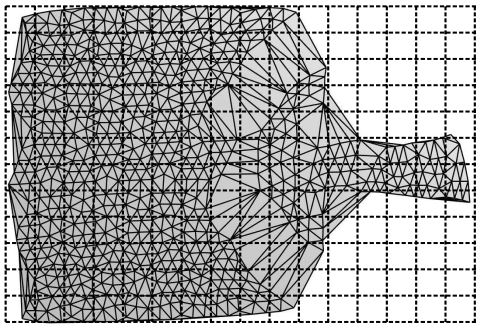
(b) Placement of the taxels obtained from the spatial calibration of the skin.

(c) The mesh S^* approximating the robot body shape S.

Figure 3.1: Steps for constructing the 3D mesh S^*.



(a) Robot tactile map M^*, obtained by flattening the 3D mesh S^*.



(b) A regular grid superimposed on M^*. Barycentric interpolation allows to compute the pressure values corresponding to the nodes of the grid.

Figure 3.2: Steps for constructing the tactile image from a 2D mesh with a non-uniform placement of the taxels.

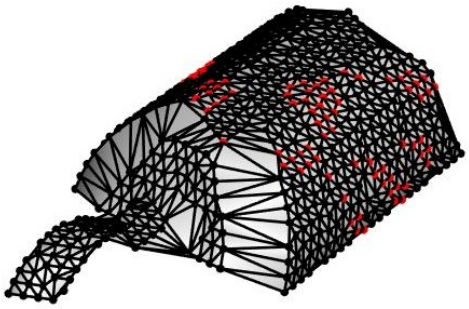
piecewise linear approximation of the robot link shape S (see Figure 3.1(c)). The idea is to exploit the *Surface Parameterization theory* (Desbrun et al., 2002) to transform the mesh S^* into a 2D *flattened* representation of the robot body, thus allowing to preserve sensor locations, displacements, density and proximity relationships among the sensors. Formally, the flattening allows to define a piecewise linear mapping $\Psi : S \rightarrow M$ between the robot body surface S and an isomorphic 2D (flat) surface M, also called *tactile map* in the following, defined by a mesh of points $M^* = (\{\mathbf{m}_1, \ldots, \mathbf{m}_N\}, F)$ where the elements $\mathbf{m}_i \in \mathbb{R}^2$ best preserve the properties of the mesh S^* minimizing the distorsions from 3D to 2D.
Therefore, for each $\mathbf{t}_i$, a corresponding $\mathbf{m}_i$ exists such that $\mathbf{t}_i = \Psi^{-1}(\mathbf{m}_i)$.
An example of the flattening transformation applied to the mesh in Figure 3.1(c) is shown in Figure 3.2(a).

It is worth noting that the computation of the map Ψ can be performed off-line for contacts expressed on a single link. Then, it does not pose significant problems for real time computations since in practice the map Ψ is implemented as a look-up table.

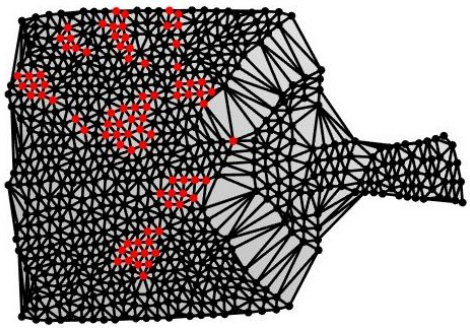
3.3 Tactile Image Creation

The tactile map M^* is a 2D entity representing the non-uniform planar displacement of the taxels. In order to generate a tactile image, M^* must be re-sampled. This is done by superimposing a regular grid with R rows and C columns on the tactile map M^*, as shown in Figure 3.2(b). The position of the grid point corresponding to row r and column c is defined as $\mathbf{x}_{rc}$.

(a) Example of a physical contact occurring on the robot forearm.



(b) Pressure measurements mapped onto the mesh S^*.



(c) Pressure measurements applied on the tactile map M^*.

(d) Resulting tactile image of the contact.

Figure 3.3: Steps for obtaining a tactile image. The actual intensity values in (b) and (c) are not shown for clarity. The taxel response is assumed to be *continuous* and not binary.

During a contact, the robot skin senses the applied pressure generating a set of measurements $P = \{p_1, p_2, \ldots, p_N\}$, where $p_i \in \mathbb{R}$ is the measurement of the i-th taxel. Figure 3.3(b) represents the discrete pressure distribution of the contact at a given time instant, obtained by associating the tactile measurements P to the mesh S^*. Similarly, P can be mapped on M^* generating a discrete *pressure map* (see Figure 3.3(c)).

In order to compute the tactile image (see Figure 3.3(d)), for each point of the grid $\mathbf{x}_{rc}$ that lies in the triangle defined by $(\mathbf{m}_j, \mathbf{m}_k, \mathbf{m}_h)$, a pressure value K_{rc} is computed using the *barycentric interpolation*, thus creating a matrix $\mathbf{K} \in \mathbb{R}^{R \times C}$ defined as:

$$\mathbf{K} = [K_{rc}] = \frac{\left(A_{kj} p_h + A_{hj} p_k + A_{hk} p_j\right)}{A}$$

where p_j, p_k and p_h are the pressure values of the taxels associated to $\mathbf{m}_j, \mathbf{m}_k, \mathbf{m}_h$, while A, A_{kj}, A_{hj} and A_{hk} are the areas of the triangles defined by the vertices $(\mathbf{m}_j, \mathbf{m}_k, \mathbf{m}_h)$,

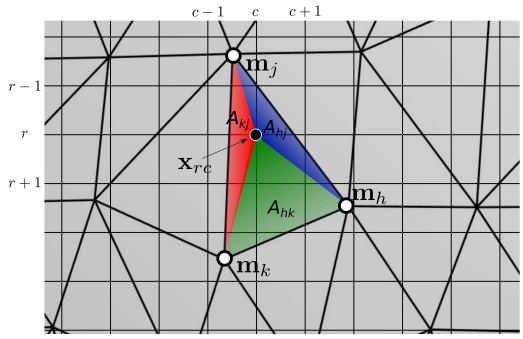


Figure 3.4: Interpolation with the pressure values of nearby vertices.

$(\mathbf{m}_j, \mathbf{m}_k, \mathbf{x}_{rc})$, $(\mathbf{m}_h, \mathbf{m}_j, \mathbf{x}_{rc})$ and $(\mathbf{m}_h, \mathbf{m}_k, \mathbf{x}_{rc})$ respectively (see Figure 3.4).

K_{rc} are the elements composing a matrix $\mathbf{K} \in \mathbb{R}^{R \times C}$ which can be converted into a classical *grayscale image*, by scaling each K_{rc} value into a grayscale level I_{rc}, using the following formula:

$$I_{rc} = 255 \left\lfloor \frac{K_{rc}}{\bar{K}} \right\rfloor \tag{3.1}$$

where $\lfloor \cdot \rfloor$ is the floor function and $i = \{1, 2, \ldots N\}$. This conversion generates an 8-bit grayscale image normalized with respect to the value $\bar{K}$, which usually corresponds to the full scale value of the robot skin. However, due to the various range of different applications exploiting tactile images, the value of $\bar{K}$ can be chosen to best suit the final task (example given in Section 4.2).
It is worth noting that the normalization above is used for the tactile image generation only, while the actual pressure exerted is known from P (or from the interpolated form $\mathbf{K}$).

3.4 Conclusion and Discussion

In this Chapter, the problem of generating a tactile image from a contact distributed on the robot body is addressed and a technique is proposed, making possible to create an image that best preserves the original 3D contact shape. The procedure described in this Chapter represents the starting point for the topic presented in Chapter 4, where tactile images are exploited in a classification task.

It is worth noting that the method is not tied to a specific technology. Indeed, as a main assumption, the method refers to a class of robot skin systems composed of discrete taxels rigidly attached to the robot links. In the literature, several examples of technologies corresponding to this assumption can be found, e.g. Cheung and Lumelsky (1989); Minato

et al. (2007); Mittendorfer and Cheng (2011); Mizuuchi et al. (2006); Mukai et al. (2008); Ohmura et al. (2006); Schmitz et al. (2011). Furthermore, the method could also be applied to other robot skin technologies not based on discrete taxel sensing, provided that the geometry of the surface is known and that the pressure at discrete points can be computed or estimated. One example of these types of tactile systems is the one based on EIT technology (Tawil et al., 2011).

Chapter 4

Human Hand Touch Recognition and Segmentation

This Chapter addresses the problem of discriminating a human voluntary touch with respect to a collision or a generic contact using robot skin measurements. The main assumption lies in the fact that, usually, a human purposive touch involves the use of hands. Therefore, machine learning techniques are combined with tactile images generated as described in Chapter 3 in order to classify the contact shape generated by a human touching the robot body with her/his hand. Moreover, it will be shown that the pressure distribution can be segmented, highlighting the role of each part of the hand involved in the contact.

4.1 Introduction and Motivation

As described in Section 2.2, the effectiveness of tactile sensors in the p-HRI domain has been proven by several authors. However, all the works discussed implicitly assumed that a person is interacting with the robot: namely, all the contacts have been generated by humans. Therefore, none of them addressed the possibility of discriminating human touch from other possible types of contacts. The goal of this Chapter is to show that such a discrimination can be achieved by analysing the shape of the contact pressure distribution.

Usually, humans physically interact with objects, or with other people, hopefully in peaceful conditions, using their hands. Similarly, in p-HRI it can be expected that if an operator wants to physically interact with a robot, for example to teach a movement (Billard et al., 2008), a natural way to begin the cooperation would be touching or grasping one or more of its links. As a matter of fact, various vision-based HRI methods are based on the assumption that the hands are the main input for interacting with robots. Indeed, they address

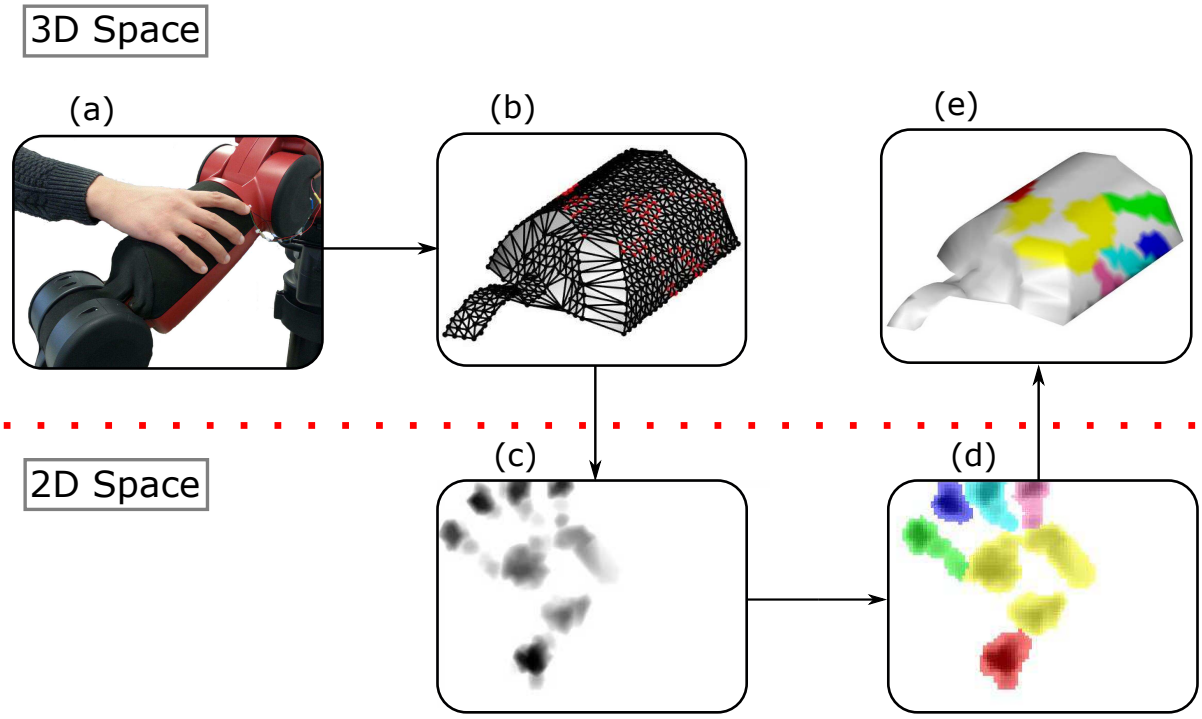


Figure 4.1: Proposed approach: (a) a human is touching the robot arm using the hand; (b) the 3D contact measurements are mapped on the mesh representing the robot body; (c) the robot skin measurements are transformed into an image and classified to recognize a human hand; (d); if it is, the parts of the hand are segmented; (e) the segmentation is back-projected on the original 3D space.

the problem of computing from images the placement of the fingers and of the palm of the human player (Liang et al., 2012; Raheja et al., 2011) in order to recognize gestures. In the p-HRI domain, it can then be argued that when a person interacts using the hand, the contact distribution generated by each finger and by the palm, in terms of positions, areas and relative applied pressures, could imply a specific type of interaction.

Therefore, according to what discussed so far, it is reasonable to assume that if a human is interacting with a robot using her/his hand, the contact could be interpreted as a voluntary touch, performed to start a cooperation. Then, in order to engage an appropriate HRI task, the robot must be capable to discriminate if the applied contact has been generated by a human and it should be capable of segmenting the measured pressure distribution associated to the various parts of the hand. This Chapter presents a method based on robot skin feedback to:

- **recognize a human voluntary touch** performed using a single hand, with respect to a generic contact or collision;

- **segment the hand contact shape**, obtaining the pressure distribution applied by each part of the hand (fingers and palm) during the interaction.

As shown in Figure 4.1, the proposed approach consists of creating a tactile image of the contact distribution by applying the transformation steps introduced in Chapter 3. Then, as explained in detail in Section 4.2, the pressure distribution will be classified and segmented using machine learning techniques since the variabilities produced by a human touching a robot skin make the definition of interaction models hard.

This Chapter is organized as follows. Section 4.2 describes the specific problems related to the processing of human hand contact shapes, motivating the usage of machine learning techniques. Sections 4.3 and 4.4 describe the machine learning-based models employed for human hand recognition and segmentation. In Section 4.5 the data collection procedure is detailed. The experimental results to asses the performance of the proposed method are discussed in Section 4.6. Conclusion and discussion follow in Section 4.7.

4.2 Tactile Images from Human Hand Contacts

As explained at the end of Section 3.3, the final step of the tactile image generation process consists in converting the interpolated grid of sensor measurements into a grayscale image, using Equation 3.1. As previously discussed, the value of $\bar{K}$ can be chosen depending on the specific operating context. In particular, since in the proposed case a shape classification task is considered, a suitable transformation that can be useful in this context is the one defined by choosing $\bar{K}$ as the maximum value applied during the current contact. The normalization of $\mathbf{K}$ allows to highlight the contact shape, making the classification and segmentation of the pressure distribution *independent* from the magnitude of the applied contact pressure.

Examples of human hand tactile images generated with the procedure discussed in Chapter 3 are shown in Figure 4.2. As it can be seen, in some images it is possible to identify the shape of the human hand, while other pictures (e.g. Figure 4.2(b), 4.2(d), 4.2(f) and 4.2(g)) can be easily confused with the non-hand contacts in Figure 4.3. However, it is quite evident that the contact shape can vary significantly even in the images where the hand is visible. For example, Figure 4.2(l) clearly shows the human hand shape, while others show just a portion of the hand or possibly only the fingertips. This is due to various factors linked to the geometry of the robot skin and to the characteristic of the interaction.

Robot skin related aspects:

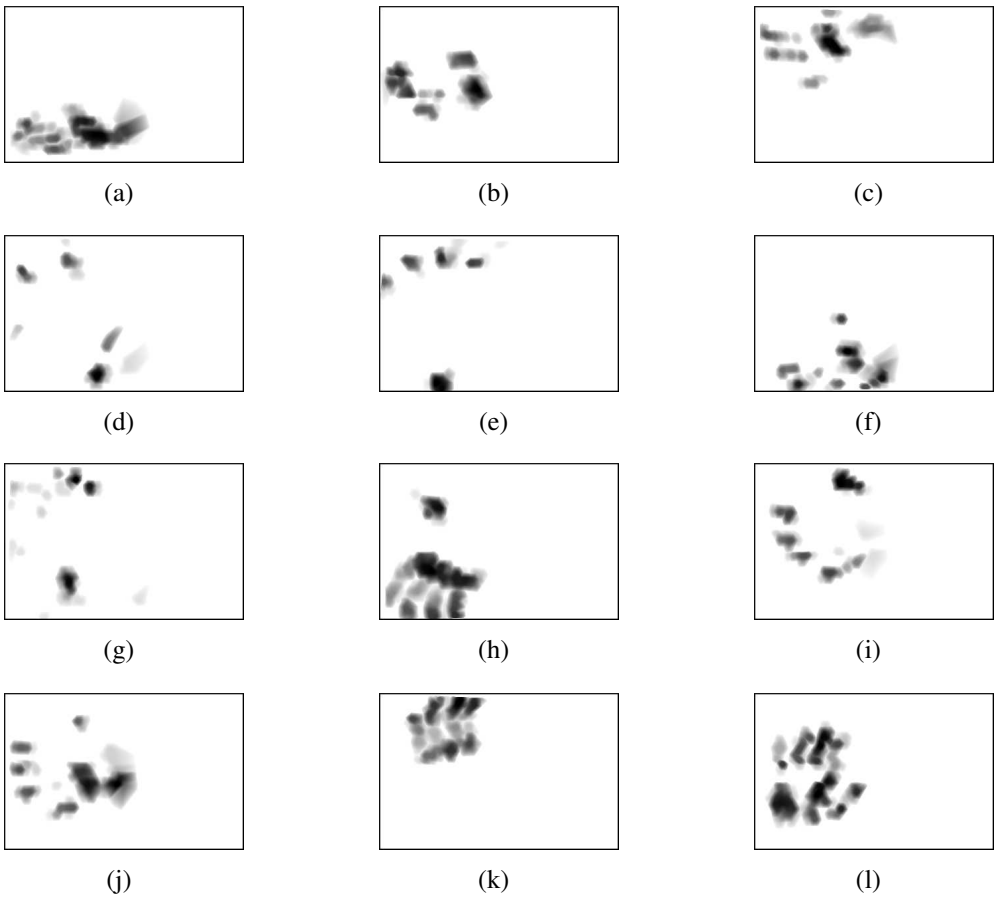


Figure 4.2: Examples of tactile images generated by human subjects during different interactions with the robot. Some fingertips seem to be cut (e.g. Figure 4.2(h)) since the person did not fully touch the sensorized area.

- unlike cameras, the spatial resolution of the tactile elements composing the skin can be not uniform. Therefore, there could be areas poorly or even not sensorized at all that could produce *holes* (loss of information) in the resulting tactile image;
- the flattening operation introduces distortions dependent on the "complexity" of the robot body shape; this implies that the similar contacts applied in different positions can produce slightly different 2D tactile images. Examples of this fact are given in Figures 4.2(h) and 4.2(k) where the fingers appear to be bent, or in Figures 4.2(a) and 4.2(f) where the distortions are more evident.

Human interaction related aspects:

- the tactile images are characterized by the type of interaction: for example, while pushing away the robot arm requires the whole hand, pulling the same part mainly

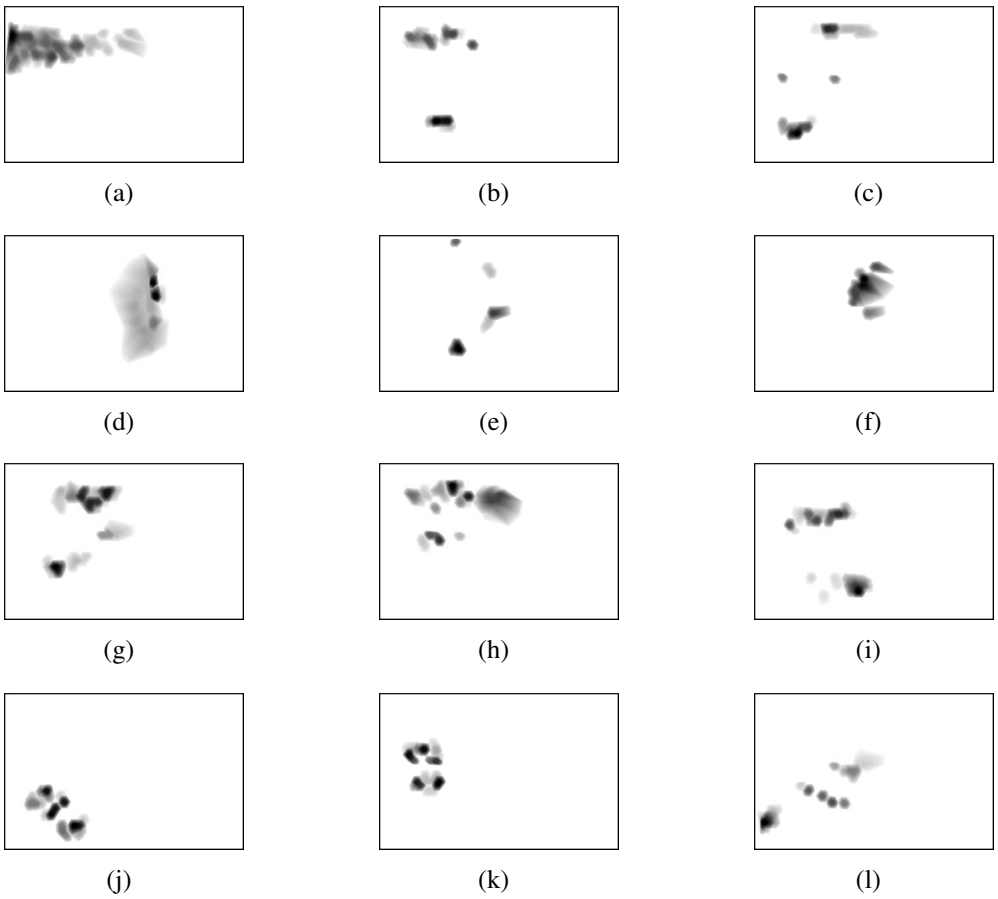


Figure 4.3: Examples of tactile images not generated by hand contacts.

involves the fingertips; moreover, in some actions not all the fingers or the palm are involved (see Figure 4.2(e) and 4.2(k));

- depending on the human operator physical characteristics (e.g. height, size of the hands, strength, etc.) and her/his relative posture with respect to the robot, each subject will interact with the robot body with different intensities or configurations of the hand; for example, Figure 4.2(i) and 4.2(j) represent a similar contact geometry expressed with different pressure distributions.

Due to these variabilities, it is hard if not impossible to define a general model of a human hand in contact with a robot body. For this reason, since our goal is to classify and segment the pressure distribution, it appears reasonable to use machine learning based techniques. In particular *supervised* methods have been considered.

4.3 Hands Classification

In order to recognize if the contact distribution is generated by a human hand, the corresponding tactile image is classified using machine learning techniques.

Convolutional Neural Networks (CNNs) for image classification outperformed previous approaches (Krizhevsky et al., 2012), proving their robustness against image variations such as scale and rotation (Farfade et al., 2015). Moreover, they have been successfully employed to recognize hand gestures in real-time (Kim et al., 2008; Lin et al., 2014; Nagi et al., 2011) and in tactile objects classification tasks (Cao et al., 2016).

When neural networks are considered, usually there are two possibilities. The first one is to define a model from scratch to solve a specific task. The second one consists in using a pre-trained model. Indeed, as discussed in Yosinski et al. (2014), when a CNN is trained on a large dataset of images, its initial layers tend to learn generic features. On the other hand, the features that are specific to a dataset are learned in the last layers. This allows to adapt a pre-trained CNN to a particular problem, by changing and training only the last part of the network.

To train a model using this methodology is the preferred option in the literature, since it speeds up the training times and it allows to use less amount of data with respect to those needed to train a network from scratch. Furthermore, pre-trained models usually show better generalization performances (Erhan et al., 2010; Guo et al., 2016).

In this work, a CNN classifier trained from scratch to recognize the human hand touch, referred in the following as **HandsNet**, is proposed. Then, since this CNN architecture is not specific for tactile measurements, but it works on images, its performance will be compared with a pre-trained model (Yosinski et al., 2014). Furthermore, since several works discussed in Section 2.2 rely on the Bag of Visual Word model (BoVW) for classifying tactile images, also the performance of this model is tested.

Table 4.1 shows the layers of the **HandsNet** model. The first part is composed of four stacked convolutional blocks, each containing three layers: a Convolutional layer with padding and stride equal to one, a Batch Normalization layer and finally a threshold operation performed through a Rectified Linear Unit (ReLU) layer (Goodfellow et al., 2016). Then the output is downsampled with a 2×2 MaxPool filter with stride 2 before being further processed.

The differences among the four blocks are in the number of filters of the convolutional layers and in the size of the kernels. According to Goodfellow et al. (2016), the depth of the network has been selected by increasing the number of layers and evaluating the accuracy on the training set, until a satisfactory performance has been obtained. The output of the last max pooling operator is sent as an input to a fully connected layer composed of 64 neurons (*fc_1*

Table 4.1: Structure of **HandsNet**. The nomenclature *conv_i* refers to a computational block formed by a convolutional layer followed by a batch normalization and finally by ReLu.

layer	**shape**
conv_1	32×7×7
max_pool_1	2×2
dropout_1 (10%)	-
conv_2	64×5×5
max_pool_2	2×2
dropout_2 (20%)	-
conv_3	128×3×3
max_pool_3	2×2
conv_4	256×3×3
max_pool_4	2×2
fc_1	64
dropout (60%)	-
fc_2	32
dropout (50%)	-
fc_3	2
softmax	2

in Table 4.1). Two further fully connected layers containing 32 and 2 neurons respectively follow. Finally, the output is a 2-way softmax unit computing a probability distribution over 2 classes: *hand* and *non-hand*.
In order to reduce the overfitting, dropout layers have been inserted by choosing their probabilities according to Park and Kwak (2017), who suggest to apply a low drop rate in the initial layers (usually < 0.5).

The classification performance of **HandsNet** has been compared with other state of the art models used in image classification. Focusing on pre-trained CNNs, there are mainly two ways to adapt a model to a particular problem. Since the initial layers of the network are able to extract generic features (Yosinski et al., 2014), one possible solution is to remove the classification layers and to use the network as a feature extractor. Once the features are computed for the new dataset, they can be used to train a new classifier (e.g. a Support Vector Machine).

The other approach is the fine-tuning, consisting in replacing the classification layer with a new one having the appropriate number of classes and then retraining the network. During this phase, the strategy is to use a very small learning rate to update the weights of the initial

layers. On the contrary, a higher learning rate is applied to train the final layers, by adapting them to the new data.

Both methods have been considered in this study applied to the VGG16 model presented in Simonyan and Zisserman (2014). This model is pre-trained on the ImageNet dataset (Deng et al., 2009), and it has been proved to be a very good choice to initialize a classifier or to be used as a feature extractor (Guo et al., 2016). Finally, the last model considered is the BoVW model, already exploited for tactile image classification.

To summarize, the four following models will be evaluated and compared:

- **HandsNet**: the model having the structure described in Table 4.1;
- **VGG16+SVM**: the features are extracted with the pre-trained VGG16 and classified using a linear SVM;
- **VGG16+ft**: fine tuning on the VGG16 pre-trained model;
- **BoVW**: Bag of Visual Word model trained with SIFT features (Lowe, 2004).

The loss function and the hyper-parameters used during the training phase are detailed in Appendix C.

4.4 Hands Segmentation

The goal of this Section is to describe how to segment the pressure distribution applied by a human hand, in order to identify the fingers and the palm area. Since tactile images are used, this task can be seen as a problem of *semantic segmentation*. Also in this case, an approach using deep learning has been considered. Indeed, the segmentation of tactile images is specific, since the number of classes could vary depending on the type of contact (e.g. the number of fingers touching the robot body could change). Furthermore, the regions composing a part of the hand could be not connected, as for the case of the palm contact in Figure 4.2(l). Therefore, the classical techniques often used in the literature (such as *k*-means, watershed, thresholds, etc.) do not appear to be suitable in this context (Dhanachandra et al., 2015; Grau et al., 2004; Morar et al., 2012).

Modern approaches presented in the past few years, dealing with the problem of semantic segmentation, rely on deep networks performing classification tasks (Guo et al., 2018), where a label is associated to each pixel instead of the whole image.
In this paper, two models have been considered: the **SegNet** (Badrinarayanan et al., 2017) and **FCN** (Long et al., 2015). Both are widely applied in the literature, representing the state of the art in semantic segmentation (Garcia-Garcia et al., 2018).

Deep networks performing a pixel-wise classification require a large amount of data to be trained from scratch. Although a dataset of human hand contacts has been collected (as detailed in the next Section), the pixel-wise classification of the whole dataset is a time-consuming operation. For this reason, the convolutional layers of both models are initialized with the weights of a VGG16 model trained on ImageNet. In this way the network can be trained using less data, thus requiring to label just a portion of the whole dataset.

The two models have been trained in order to segment and recognize the following 6 classes: *Thumb, Index, Middle, Ring, Pinkie* and *Palm*. The training details are reported in Appendix C.

4.5 Dataset

As discussed in Section 4.2, the contact shape of the human hand depends on several factors. In order to properly train the models described in Sections 4.3 and 4.4 there is the need for a dataset capturing as much variabilities in the contact shape as possible. To this aim, the data collection procedure has been designed to capture the interactions between the human and the robot in different conditions.

The dataset has been collected performing an experiment which involved voluntary human subjects [1]. People have been asked to interact with a Baxter robot arm partially covered with robot skin. The robot skin and the experimental platform used in this experiment are respectively described in detail in Appendices A and B. The subjects performed the following actions, which correspond to *natural* interaction modes in p-HRI:

1. Grasp the forearm;
2. Grasp and torque clockwise the forearm (i.e. a twist with respect to the forearm axis);
3. Grasp and torque counter-clockwise the forearm;
4. Push the forearm to the left;
5. Push the forearm to the right;
6. Push the forearm away;
7. Pull the forearm;

[01] Each subject signed an informed consent form and all the data have been carefully anonymized.

(a) A user stands in the front of the robot.

(b) A user stands on the side of the robot.

Figure 4.4: Two different positions taken by a human during the experiments: in front of the robot (a) and on its side (b).

Each action has been repeated twice in two different positions of the robot arm (see Figure 4.4). Each person interacted with the robot without any constraint related to the hand posture and intensity of the touch. After that, for five times, the user moved the robot arm in to a different configuration, performing one interaction of the list. In this phase, the arm position, the relative posture with respect to the robot and the interaction type have been chosen by the user.

During the whole experiment, the robot is commanded to maintain its pose and the entire interaction has been recorded. Each interaction produced a sequence of samples consisting of sensors measurements collected with a sampling time of 0.1 seconds. From this sequence, the sample with the highest number of taxels activated by the contact is selected to generate a single tactile image, as described in Chapter 3.
The tactile images have been generated using a regular grid with a step size of 1 *mm*. The robot tactile map (see Figure 3.2(a)) has a dimension of 247 *mm*$\times$ 362 *mm*, so the corresponding tactile image is composed of 247 $\times$ 362 pixels. Finally, in order to reduce the noise and further highlight the contact shape, an erosion followed by a dilatation of the image have been performed (Beyerer et al., 2016), using a circular structural element with 2 and 4 pixels of radius respectively.

The number of people involved in the experiment is 90. The subjects have different gender (66.67% Male, 33.33% Female), handedness (77.78% Right, 22.22% Left) and biometric characteristics (Table 4.2). At the end of the data collection, **1710** tactile images of hands have been acquired.

Table 4.2: Summary of the characteristics of the subjects involved in the experiment.

	Hand Length [2]	**Age**	**Weight**	**Height**
Min	15 cm	20	48 Kg	154 cm
Max	22 cm	59	105 Kg	194 cm
Mean	18 cm	26	70 Kg	178 cm

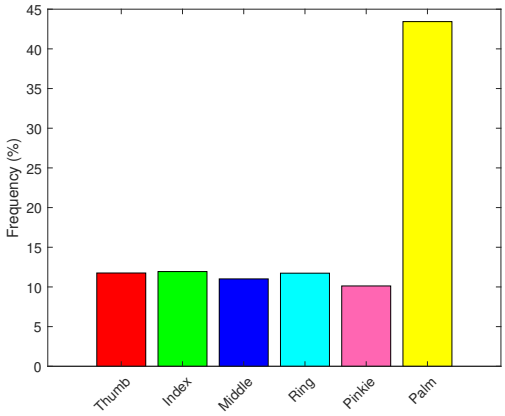


Figure 4.5: Histogram representing the average frequency of pixels for segmented class. The colors shown in the histogram are also used in the following to identify the segments in the tactile images.

In order to train the models described in Section 5, the dataset has been completed by adding **1820** *non-hand* images produced from contacts with other human limbs or generic objects. Contacts with objects have been collected by the authors over time by touching the robot on the sensorized area with objects having different properties such as shape, size, material (e.g. plastic, metal, etc.) and softness. The contacts with human body parts (e.g. torso, arm, forearm, shoulder, back) have been collected both by the author and by the subjects involved in the experiment without using a formal protocol. In particular, all the users have been asked to touch the robot five times with different body parts other than the hand. In summary, about 35% non-hand images have been created from contacts with body parts and the remainder from contacts with objects.

Some examples are shown in Figure 4.3. As an outcome, the dataset used to train the classifiers in Section 5 is composed of **3530** tactile images.
The dataset has been split in a training set (70%) and in a test set (30%). In order to evaluate the classifiers on previously unseen human subjects, the test set has been created containing images generated from subjects not included in the training set.

[02] The hand length is measured from the wrist to the tip of the middle finger.

The semantic segmentation models described in Section 6 require pixel-wise labelled tactile images as ground truth. According to what discussed in Section 6, the initialization with pre-trained weights allowed to use only a fraction of the whole dataset. In particular, **350** samples have been picked from the whole dataset of human hands and labelled pixel by pixel. The distribution of classes is shown in Figure 4.5. Also for this task, the dataset has been split in training set (70%) and test set (30%).

Both datasets, for the classification and segmentation tasks, are provided as supplementary material.

4.6 Experimental Results

This Section reports the experimental results obtained with the models previously described using the datasets acquired as discussed in Section 4.5.
The models have been trained on Matlab running on a server equipped with two Intel Xeon E5 CPUs and two Nvidia P100 GPUs with 16 GB of RAM each.
For each model, a set of hyper-parameters has been selected and tuned. Details about the training and tuning procedures are reported in Appendix C.

4.6.1 Human Hand Touch Classification

The models trained with the parameters described in Appendix C are evaluated on the test set. The results are shown in Table 4.3, where the mean accuracy and the classification times are reported.

Table 4.3: Performance of the models. For each one the mean accuracy on the test set and the time for classifying one tactile image have been computed.

	Accuracy	**Time (ms)**
HandsNet	97.81%	12.6
VGG16+SVM	95.40%	14.4
VGG16+ft	96.69%	27.5
BoVW	94.03%	17.6

A more detailed analysis about the results obtained on the test set is shown in Tables 4.4-4.7, representing the confusion matrices of the models. It can be seen that **HandsNet** performs slightly better than **VGG16+ft**. The difference in terms of accuracy is larger than 1% and it is faster with respect to **VGG16+ft**. It is worth noting that the model **VGG16+SVM** obtained good results in terms of accuracy and time, having only a single hyper-parameter

Table 4.4: Confusion matrix of the **HandsNet** model applied on the test set. The mean accuracy is 97.81%.

	Hand	**Non-Hand**
Hand	96.88%	1.28%
Non-Hand	3.12%	98.72%

Table 4.5: Confusion matrix of the **VGG16+SVM** model applied on the test set. The mean accuracy is 95.40%.

	Hand	**Non-Hand**
Hand	96.49%	5.69%
Non-Hand	3.51%	94.31%

to tune (see Appendix C), while the **BoVW** showed lower performance with respect to the other models.

Table 4.6: Confusion matrix of the **VGG16+ft** model applied on the test set. The mean accuracy is 96.75%.

	Hand	**Non-Hand**
Hand	98.64%	5.14%
Non-Hand	1.36%	94.86%

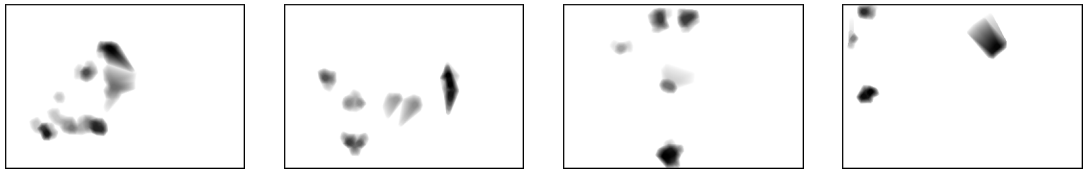
Table 4.7: Confusion matrix of **BoVW** classifier applied on the test set. The mean accuracy is 94.03%.

	Hand	**Non-Hand**
Hand	96.49%	8.44%
Non-Hand	3.51%	91.56%

An example of tactile images misclassified by the **HandsNet** model is given in Figure 4.6, while the full list of tactile images correctly classified and misclassified for each model can be found in the provided supplementary material.

4.6.2 Human Hand Touch Segmentation

To evaluate the models described in Section 6, the four metrics discussed in Long et al. (2015) have been considered. The first one is the *pixel accuracy* **Acc**, which evaluates the percentage of correctly classified pixels without considering their classes. The second is the *pixel mean accuracy* **mAcc**, i.e. the percentage of correctly predicted pixels for each class,



(a) (b) (c) (d)

Figure 4.6: Examples of tactile images misclassified by **HandsNet**; (a) and (b): non-human hand contacts classified as hands; (c) and (d): human hand contacts classified as non-hands.

averaged over the classes.
The third metric is the *mean intersection over union* **mIoU**, which computes how well the sets of predicted classes overlap the ground truth. Finally, due to the presence of imbalances in the dataset (see Figure 4.5), the *frequency weighted intersection over union* **fwIoU** has also been considered, i.e. a weighted version of the mIoU which takes into account the appearance frequency of each class.

Table 4.8 reports the scores obtained on the test set for each metric. **SegNet** model outperforms **FCN** providing also a lower inference time. The confusion matrices in Table 4.9 and 4.10 give a detailed information about the pixel accuracy for each class. A comparative example between the two models is shown in Figure 4.7.

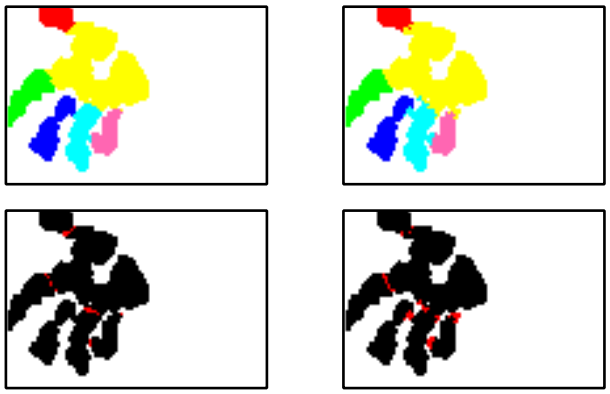
Table 4.8: Metrics evaluated for both models on the test set.

	Acc	**mAcc**	**mIoU**	**fwIoU**	**Time (ms)**
SegNet	93.37%	93.05%	89.17%	90.53%	63.37
FCN	88.82%	85.69%	80.14%	83.06%	75.13

Table 4.9: Confusion matrix of the **SegNet** model fed with the test set.

	Thumb	**Index**	**Middle**	**Ring**	**Pinkie**	**Palm**
Thumb	95.05%	0%	0%	0%	0.04%	0.92%
Index	0%	92.85%	2.59%	0.60%	2.10%	0.92%
Middle	0%	1.29%	90.34%	3.88%	0.25%	0.10%
Ring	0%	0.31%	5.07%	92.08%	1.03%	0.42%
Pinkie	0.38%	2.21 %	0.43%	2.56%	91.34%	1.02%
Palm	4.47%	3.33%	1.57%	0.88%	5.20%	96.61%

Focusing on **SegNet**, Figure 4.8(a) to 4.8(j) show a set of segmented tactile images (first row), along with the misclassified pixels (second row). As it can be seen, the network



(a) SegNet output **mAcc**: 98.77%.

(b) FCN output **mAcc**: 94.86%.

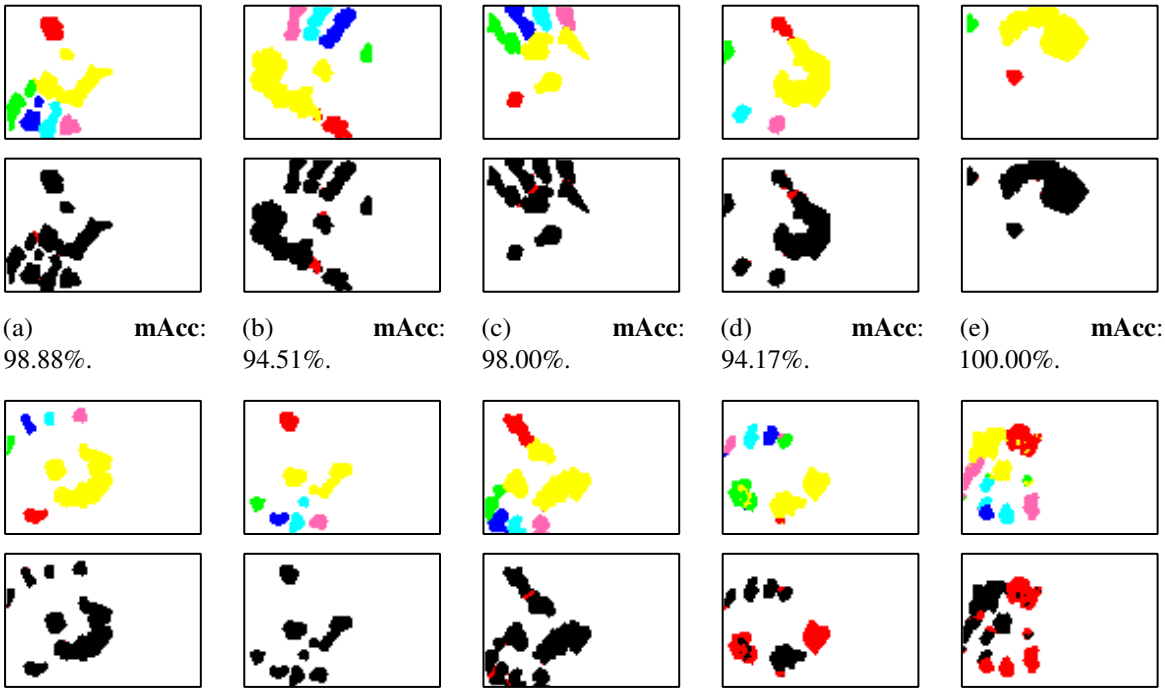
Figure 4.7: Segmentation performed by **SegNet** and **FCN** on the same tactile image. The first line shows the models output. The colors of the various segments are the same used in Figure 4.5. The second line shows the tactile image in binary scale with red pixels corresponding to misclassified regions.

Table 4.10: Confusion matrix of the **FCN** model fed with the test set.

	Thumb	**Index**	**Middle**	**Ring**	**Pinkie**	**Palm**
Thumb	91.24%	0.75%	0.31%	0.23%	1.12%	0.90%
Index	0.81%	83.95%	3.31%	0.40%	3.51%	0.98%
Middle	0%	3.09%	78.41%	5.92%	1.30%	0.61%
Ring	0%	0.79%	9.91%	84.47%	1.48%	0.61%
Pinkie	0.6%	3.12%	1.62%	2.41%	80.10%	0.93%
Palm	7.88%	8.28%	6.42%	6.55%	12.47%	95.96%

is able to correctly create the clusters under different conditions. For example in Figure 4.8(a) and 4.8(b) almost the whole hand is in contact with the robot body. On the contrary, Figure 4.8(c), 4.8(d) and 4.8(e) show contacts where the fingers or the palm are partially or completely not involved. The network can also correctly segment fingers composed of non-connected regions as visible in Figure 4.8(f) and 4.8(g), or when the fingers are bent due to the distortions introduced by the flattening (see Figure 4.8(h)). Instead, Figure 4.8(i) and 4.8(j) show two examples of poorly segmented tactile images with a mean pixel accuracy lower than 80%.

The full list of images segmented using both models is included as supplementary material.



(a) **mAcc**: 98.88%. (b) **mAcc**: 94.51%. (c) **mAcc**: 98.00%. (d) **mAcc**: 94.17%. (e) **mAcc**: 100.00%.

(f) **mAcc**: 100.00%. (g) **mAcc**: 100.00%. (h) **mAcc**: 96.87%. (i) **mAcc**: 72.21%. (j) **mAcc**: 41.51%.

Figure 4.8: Examples of segmentation results. First line: SegNet output. Second line: thresholded tactile image with red areas corresponding to misclassified pixels.

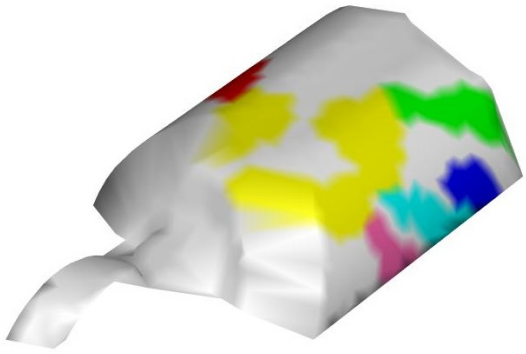
4.7 Conclusion and Discussion

In this Chapter, a technique allowing to discriminate between human hand contacts and other generic types of contacts has been proposed. Furthermore, it has been shown that human hand contacts can be segmented with a good accuracy to recognize the various hand parts involved in the contact.

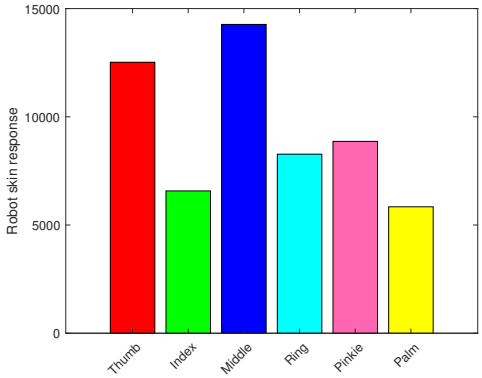
With respect to the existing literature, mostly based on the processing of planar tactile measurements, the set of transformations described in Chapter 3 is applied here to process robot skin measurements, thus leading to a 2D tactile image which can be processed and classified using state of the art image processing techniques.

The results of this Chapter can have a major impact in the domain of p-HRI since the recognition of a human hand contact can be seen as a voluntary interaction aimed at starting a cooperation. Moreover, the possibility of segmenting the pressure distribution can provide relevant information about the role of the various parts of the hand involved in the interaction. An example is given in Figure 4.9, where it can be seen that, after the segmentation operation,

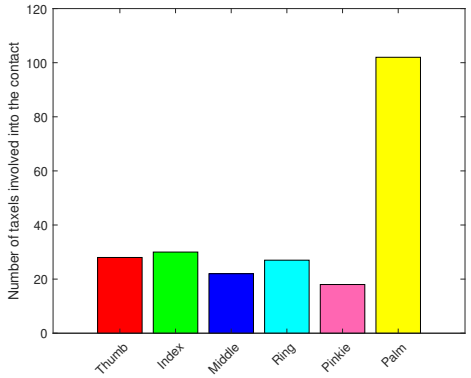
(a) Original tactile image.



(b) Segmented areas mapped on the 3D model of the robot.



(c) Mean pressure distribution for each part of the hand. The values are related to the robot skin raw output, which can range from 0 to 65536 (see Appendix A).



(d) Number of taxels involved in the contact for each part of the hand.

Figure 4.9: The segmentation is used for retrieving contact properties for each part of the hand involved in the interaction.

the information related to the contact distribution can be extracted for each part of the hand involved in the contact.

The models used in the classification tasks have been implemented using Matlab 2018b, with acceptable time performance with respect to the sampling rate of the tactile images. This suggests that an efficient implementation of the models, using optimized libraries, such as Tensorflow (Abadi et al., 2015), can further speed-up the computation.

The results presented in this Chapter represent the stand point for further research. Firstly by considering the problem of multiple contacts. Secondly, addressing the problem of recognizing the type of physical human interaction (e.g. Push, Pull, Twist, etc.) by analysing the contact dynamics considering sequences of tactile images.

Chapter 5

Robustness and Transferability Analysis

The method proposed in the previous Chapter, to classify and segment contact shapes generated by human hands, has been tested considering contacts occurred on the same robot link. This Chapter extends the concept introduced in Chapter 4 providing additional experiments, considering tactile images generated from: (i) tactile maps with different spatial resolution; (ii) different robot body parts. Changes in the spatial resolution of the skin can be interpreted as the response of skins with different taxel distributions as well as the effect of variations of the robot skin response due to possibly faulty taxels. In summary, the content proposed in this Chapter allows to perform an analysis from two different points of view: against possible hardware faults and with respect to different taxel distributions.

5.1 Introduction and Motivation

The contents described in Chapters 3 and 4 introduced a *pipeline*, allowing to perform tactile classification tasks starting from raw measurements. In the dataset discussed in Section 4.5, tactile images are generated from the same hardware and the parameters of the models have been tuned to optimally work with a fixed robot skin geometry. Therefore, if the images are generated from a tactile system with *different taxels distribution*, a loss of performance of the task would be expected.

In this Chapter, three experiments are proposed to validate the performance of classifiers trained on a specific robot geometry and fed with images generated from a robot skin having different spatial distribution. The classification/segmentation problem described in Chapter 4 is used as a reference to evaluate the robustness of the whole pipeline composed by the tactile measurement transformation and the final classification task (see Figure 4.1). In particular, two operative conditions are evaluated: (i) tactile images generated from the same geometry

but with different spatial resolution; (ii) tactile images generated from a completely different robot part.

To analyse the loss of performance with respect to changes in the spatial resolution is useful for two aspects. Firstly, since to design a robot skin with high resolution increases costs and the complexity, analysing how specific tasks perform varying the spatial resolution can provide useful guidelines to optimally design the hardware. Secondly, and most importantly, changes in the spatial resolution of the skin can be also interpreted as the effect of variations in the robot skin response due to hardware *failures*. Indeed, due to repeated physical contacts, the elements composing a robot skin can be damaged. When faulty sensors are detected they can be removed from the tactile map, producing a new map with *lower* resolution.

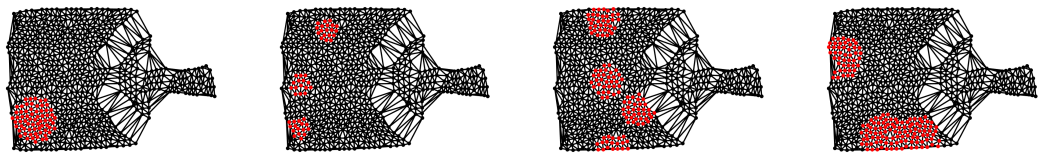
Failures are a critical aspect, since the complexity and the costs of the system could make difficult or infeasible to replace a damaged part. For example, the sensorization of the Baxter robot described in Appendix B took months. As a matter of fact, the substitution of a broken part should be the last resort. Therefore, an analysis of the performance of the task considering tactile images generated from the same tactile map, but with different resolution, becomes relevant. In particular, two different types of failures have been considered. In the first case, it is assumed that one or more groups of contiguous taxels fail during a physical interaction, causing a set of *blind spots* in the tactile image.
In the second case, the analysis is made assuming to eliminate a random distribution of faulty taxels (likewise a salt and pepper noise) from the 2D triangulation, producing a tactile map with *lower* spatial resolution. To this aim, two experimental tests have been conducted, simulating: (i) failures of groups of taxels (**Test A**); (ii) randomly distributed faulty taxels (**Test B**). A third experiment (**Test C**), preliminarily analyses the transferability of the classification model, evaluating it on tactile images generated from contacts occurring on a robot link having a significantly different geometry.

In order to benchmark these experiments we used the models **HandsNet** and **SegNet** for the classification and segmentation task respectively, which best performed in Section 4.6.

5.2 Test A

The goal of this experiment is to evaluate the performance of the proposed method when groups of contiguous tactile elements stop working, possibly at run time. In this scenario, it is assumed that the response of the faulty taxels is zero, producing a sort of *blind spot* in the tactile map. The problem of detecting faulty taxels and to set the corresponding measurements to zero is part of the data acquisition and the processing pipeline and it is beyond the scope of this Chapter.

Several tactile maps affected by randomly generated patterns of faulty taxels (i.e. *corrupted maps*) have been considered. For each contact, corresponding to images belonging to the test sets described in Section 4.5, a new tactile image has been regenerated using the corrupted map for both the classification and segmentation tasks. Then, the performance of the models has been evaluated on these new test sets of images. The failure patterns have been created using the following procedure: a taxel lying on the tactile map is randomly selected as the center of the blind spot, then the response of all the taxels within a distance of $\bar{r}$ is set to zero. The number of blind spots N_s corrupting a tactile map can range from 1 to 4, while the radius of the spots $\bar{r}$ varies from 10 *mm* to 40 *mm* with steps of 10 *mm*. For each one of the 16 combinations of these parameters, 10 random patterns have been generated, leading to a total of 160 corrupted maps. Examples of corrupted maps with different values of N_s and $\bar{r}$ are shown in Figure 5.1. The full list of corrupted tactile maps is included as a supplementary material.



(a) $\boldsymbol{N_s} = 1$ and $\bar{\boldsymbol{r}} = 40$ (b) $\boldsymbol{N_s} = 3$ and $\bar{\boldsymbol{r}} = 20$ (c) $\boldsymbol{N_s} = 4$ and $\bar{\boldsymbol{r}} = 30$ (d) $\boldsymbol{N_s} = 3$ and $\bar{\boldsymbol{r}} = 40$

Figure 5.1: Examples of corrupted tactile maps. Red areas correspond to contiguous regions of faulty taxels.

In order to evaluate the performance in the case of the segmentation task, the same blind spots appearing on the test images have been transferred to the ground truth images.

Tables 5.1 and 5.2 show the performance for each combination of N_s and $\bar{r}$, computed by averaging the results obtained for the corresponding 10 random patterns. From Tables 5.1 and 5.2 it can be seen that in the classification case the system provides an acceptable performance even with high levels of degradation. In the case of the segmentation task, the proposed method is less robust, providing a mean accuracy of about 80% in the worst case.

5.3 Test B

After a failure is detected and there is no contact occurring, the faulty taxels can be removed from the tactile map and the triangulation can be recomputed, thus generating a tactile map with *lower* spatial resolution. In this experiment, a salt and pepper faulty pattern is simulated, randomly removing from the tactile map a certain percentage $\bar{p}$ of taxels. The goal is to

Table 5.1: **Test A - Classification:** Mean scores obtained over the 10 test sets for each combination of number of spots and radius values.

N_s	$\bar{r}$ (mm)	**Accuracy**
1	10	97.81%
1	20	97.50%
1	30	97.61%
1	40	97.21%
2	10	97.68%
2	20	97.28%
2	30	96.88%
2	40	95.18%
3	10	97.73%
3	20	97.00%
3	30	96.16%
3	40	93.95%
4	10	97.67%
4	20	96.78%
4	30	94.42%
4	40	89.74%

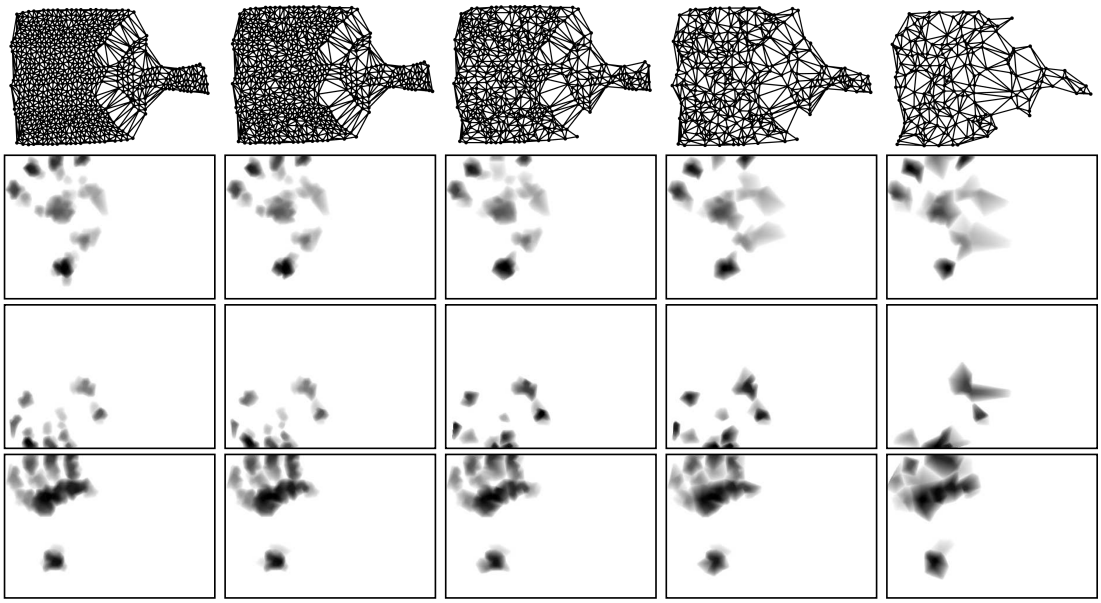
Table 5.2: **Test A - Segmentation:** Mean scores obtained over the 10 test sets for each combination of number of spots and radius values.

N_s	r	**Acc**	**mAcc**	**mIoU**	**fwIoU**
1	10	93.18%	92.58%	88.66%	90.33%
1	20	92.81%	92.07%	88.03%	89.91%
1	30	92.48%	91.33%	87.36%	89.59%
1	40	91.53%	90.03%	85.89%	88.58%
2	10	93.14%	92.56%	88.63%	90.28%
2	20	92.61%	91.76%	87.72%	89.67%
2	30	91.70%	90.40%	86.04%	88.65%
2	40	90.22%	88.63%	83.96%	87.06%
3	10	93.03%	92.42%	88.44%	90.16%
3	20	91.81%	90.66%	86.05%	88.68%
3	30	90.05%	88.05%	83.43%	86.90%
3	40	86.42%	82.47%	77.51%	83.05%
4	10	92.90%	92.27%	88.31%	90.03%
4	20	92.01%	90.99%	86.81%	89.06%
4	30	88.76%	85.89%	81.03%	85.42%
4	40	84.15%	80.20%	74.40%	80.52%

benchmark the system, evaluating its dependency on the spatial resolution of the tactile map. The percentage of removed taxels $\bar{p}$ is a parameter which varies from 10% to 70% with steps of 10%. Taxels are *incrementally* removed. This means that the taxels lying on the tactile map generated with 20% of faulty sensors are a subset of the ones generated with 10%.

Once the taxels are removed from the tactile map, the triangulation is recomputed. Also in this case, 10 patterns of broken sensors are randomly generated for each percentage value; therefore, 70 different tactile maps have been created and for each one a corresponding dataset of tactile images has been generated. Figure 5.2 shows examples of the degradation obtained for different percentage of removed taxels. The full list of downsampled tactile maps is included as a supplementary material.

The benchmark for the segmentation task requires labeled ground truth images (see Section 4.5). Since the tactile maps have changed, to exactly evaluate the performance of the segmentation model it would require to label pixel-wise all the 70 dataset of tactile images: this is practically an infeasible operation. In order to overcome this issue, for each low resolution tactile images, the following procedure has been applied. Given $\mathbf{I}_H^T$ the segmented



(a) Original (b) $\bar{\mathbf{p}} = 10\%$ (c) $\bar{\mathbf{p}} = 30\%$ (d) $\bar{\mathbf{p}} = 50\%$ (e) $\bar{\mathbf{p}} = 70\%$

Figure 5.2: Examples of downsampled tactile maps and tactile images generated for different values of $\bar{\mathbf{p}}$. The first row shows the tactile maps, while the remaining rows show the level of degradation of the tactile images generated from the corresponding tactile map.

ground truth image at full resolution (see Section 4.5), and given $\mathbf{I}_L^O$ the corresponding tactile image generated from a low resolution map, a binary mask is computed as:

$$\mathbf{I}_M = \left[\mathbf{I}_H^T\right]^B \wedge \left[\mathbf{I}_L^O\right]^B$$

where $[\cdot]^B$ is the thresholding operator and $\wedge$ is the logical AND operator. Then the actual low resolution pair $\left(\mathbf{I}_L, \mathbf{I}_L^T\right)$ is computed as:

$$\mathbf{I}_L = \mathbf{I}_L^O \circ \mathbf{I}_M$$
$$\mathbf{I}_L^T = \mathbf{I}_H^T \circ \mathbf{I}_M$$

where $\circ$ represents the pixel-wise product. Figure 5.3 graphically describes this process. Clearly, this is an approximation, since part of the pixels is not considered. However, it gives a *qualitative* assessment of the results obtained when lowering the resolution of the tactile map.

Tables 5.3 and 5.4 show the accuracy of the models described in Sections 4.3 and 4.4, evaluated on the low resolution test sets. Similarly to **Test A**, the scores are computed by averaging the results obtained on the 10 datasets generated for each $\bar{p}$ value. In Table 5.4,

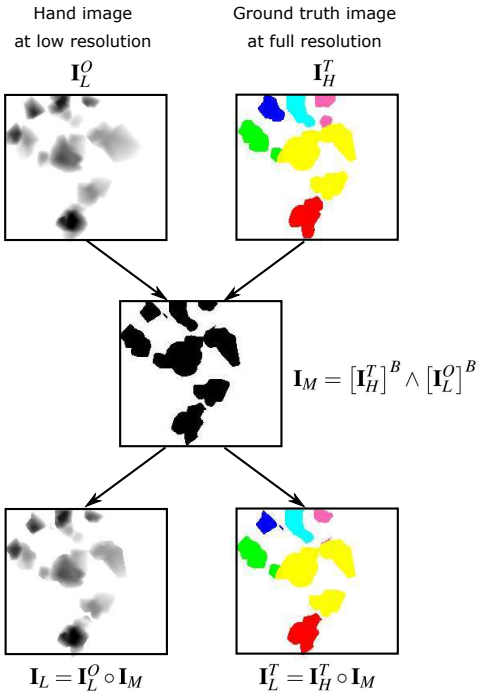


Figure 5.3: Process for generating data to evaluate the segmentation model with low resolution tactile maps. In the example, the hand image is generated from a tactile map where 40% of the taxels have been removed.

the quantity p_d represents the mean percentage of pixels discarded from the low resolution image as a result of the masking operation described above.

Table 5.3: **Test B - Classification:** Mean scores obtained over the 10 test sets for each value of $\bar{p}$.

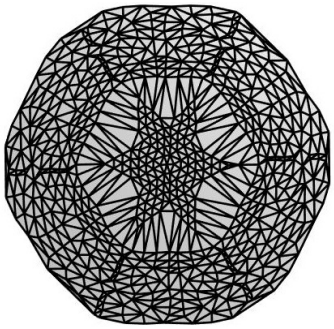
$\bar{p}$	Accuracy
10%	97.33%
20%	96.98%
30%	96.63%
40%	95.93%
50%	94.79%
60%	92.67%
70%	88.67%

Table 5.4: **Test B - Segmentation:** Mean scores obtained over the 10 test sets for each value of $\bar{p}$.

$\bar{p}$	Acc	mAcc	mIoU	fwIoU	p_d
10%	92.75%	91.69%	87.83%	89.85%	6.05%
20%	92.59%	91.16%	87.31%	89.65%	11.47%
30%	92.17%	90.54%	86.62%	89.21%	16.40%
40%	91.77%	89.55%	85.69%	88.79%	20.88%
50%	90.71%	87.62%	83.48%	87.43%	25.41%
60%	89.13%	84.88%	80.23%	85.54%	30.58%
70%	84.83%	78.20%	72.83%	80.84%	35.87%

The results obtained from this experiment show that the system is robust with respect to changes in spatial resolution of the sensors. Indeed, even with 60% of taxels removed, the

(a) Robot end-effector partially covered with tactile sensors.



(b) The robot end-effector tactile map.

(c) Example of a tactile images generated by a human touching the end-effector.

Figure 5.4: The sensorized robot end-effector used in this experiment.

system provides a classification accuracy above 90%. In the case of the segmentation task, a mean accuracy higher than 90% can be achieved considering 30% of faulty taxels.

5.4 Test C

This experiment preliminarily analyses if a model trained on a particular robot geometry can be transferred to another robot link. The results obtained with Test A and B suggest that the system has a low sensitivity with respect to small variations on the input. Therefore, it is reasonable to expect that the models applied on tactile images generated from robot surfaces having a similar geometry should provide acceptable performance.

On the contrary, in this context, a robot link with a very different shape to the one used in Chapter 4 is considered. The new part, consisting in a robot end-effector, is shown in Figure 5.4, along with its tactile map and an example of a tactile image generated from a human hand contact. As it can be seen, the contacts on this tactile map are mapped generating tactile images completely different from the ones used for training the models in Chapter 4.
Considering the classification task only, a new dataset is required to validate the **HandsNet** model on this new geometry. The end-effector has been attached to the robot and a new dataset has been collected following the same procedure described in Section 7.2. These new experiments involved 12 people, leading to a new dataset composed of 228 hand images and 250 non-hand images.

A first experiment consisted in feeding the model considering the whole amount of images as a new test set. This produced very poor results, with a mean accuracy below 53%. As it can be seen from Table 5.5, almost all the hand contacts are misclassified, which is

Table 5.5: Confusion matrix of the **HandsNet** model fed with the images generated from the robot end-effector tactile map. The mean accuracy is 52.92%.

	Hand	**Non-Hand**
Hand	8.71%	2.87%
Non-Hand	91.29%	97.13%

reasonable, since the human hand shape is mapped in a completely different way with respect to the original case.

Table 5.6: Confusion matrix of the **HandsNet** model after the fine-tuning procedure. Results are computed on the test set of tactile images generated from contacts occurring on the robot end-effector. The mean accuracy is 93.57% .

	Hand	**Non-Hand**
Hand	92.41%	5.26%
Non-Hand	7.59%	94.74%

A possible solution to get better results would be performing a fine tuning, allowing the model to learn the new introduced distortions. So the new dataset has been split into training and test sets using the same modalities described in Section 7.2. Then a fine-tuning of the **HandsNet** model has been performed using the train set. The model has been trained on the new data for 120 epochs using a batch size of 128 and a learning rate of 0.01 which has been halved after 60 epochs. The learning rate applied during the training has been reduced of a 0.1 factor in the first two convolutional layers. The training process led to a mean accuracy higher than 93%. In this phase an intensive hyper-parameters tuning procedure has not been performed. Table 5.6 shows the confusion matrix of the model fed with the test set.

5.5 Conclusion and Discussion

In this Chapter, a robustness and transferability analysis has been performed considering the pipeline introduced in Chapters 3 and 4. It is worth noting that to the best of the author's knowledge the problem addressed in this Chapter has not been considered yet in the literature of tactile classification.

Results suggest that the proposed touch classification and segmentation procedures are robust with respect to hardware failures as well as to changes in the spatial resolution. For what concerns the transferability analysis, results show that, even considering very different robot body geometries, the fine tuning allows to adapt the model. Although in this case the

performance are lower with respect to the original case, the accuracy is high enough to apply the model in a real word scenario. However this is a preliminary result. The whole problem of the transferability requires a deeper study. This experiment just showed that the original model trained with tactile images can be transferred with acceptable performances to another robot surface with a significantly different shape. There are still open questions. For example, an increased number of data or a better choice of the parameters used in the tuning procedure could improve the results. Another solution would be to perform different types of cuts on the robot skin mesh during the flattening operation, thus introducing distortions more similar to the original tactile map used for training the models.

Chapter 6

Towards a 2D Whole Robot Body Representation

When tactile sensors cover the whole robot body, their relative spatial relations change depending on the robot posture, making the definition of a 2D representation of the robot body non trivial. When contacts occur on multiple links or across two adjacent links, the current relations among the tactile elements must be considered in order to generate a tactile image that preserves the contact shape. This Chapter preliminarily addresses the problem.

6.1 Introduction and Motivation

Chapter 3 shows that tactile maps can be used as a robot internal representation for generating tactile images best preserving the contact shape. So far, the proposed method has been defined for tactile robot skin patches rigidly attached to a single robot link. The generalization of the procedure considering the whole robot kinematic chain is not trivial. Indeed, while the robot is moving, the relative spatial relations among the taxels belonging to different links change accordingly. In order to create a tactile image that best preserves the contact shape in case of contacts on multiple links or across two links, the changes in the spatial relations should be properly reflected on the corresponding tactile map of the robot body.

The goal of this Chapter is to preliminarily address the problem of constructing a tactile map of the whole robot body. The results presented in this Chapter are not conclusive; only the case of two connected joints is considered. However, the concepts developed in the following are useful to frame the general problem and they can be used as a starting point for a further extension of the method considering an arbitrary number of links.

When designing a method allowing to generate a 2D representation of the robot body while the spatial relations among the taxels are changing, some aspects have to be taken into account.
One is related to the *computational time*. The flattening is an intensive operation from the computational point of view. Approaches that require to flatten the entire robot skin at each time instant are not computationally feasible. Furthermore, the more complex is the surface to be flattened, the more difficult is to preserve the relations among the vertices composing the mesh.
A possible solution to overcome these issues would be to perform an on-line flattening of the *contact surface* only, thus creating smaller tactile maps at each time instant. This approach could be computationally feasible since the surface to be flattened contains only the taxels belonging to the contact area, which are just a small fraction of the entire robot skin. However, this solution has two major disadvantages:

- since the contact surface can significantly change during time, the introduced distortions are different for each generated tactile map. This could represent a problem, for example when tactile images have to be classified. Contact shapes generated by the same classes of objects can be distorted in different ways, making a classification task more difficult;

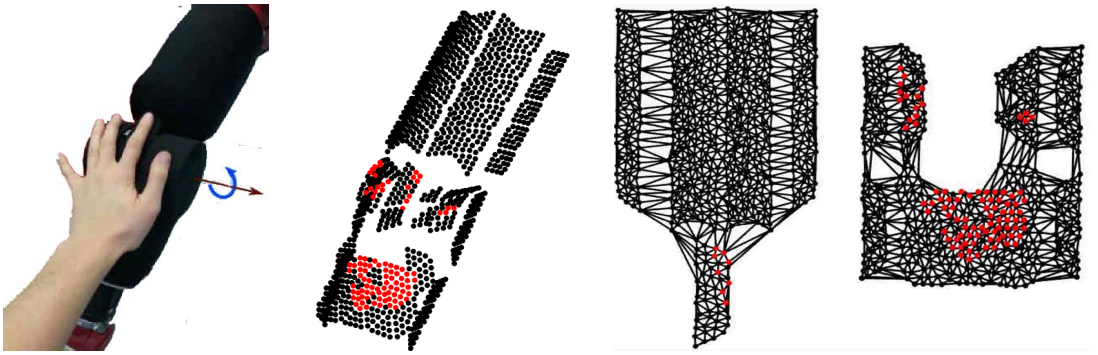
- the computational time varies depending on the size of the contact area. Therefore, the time required to perform the flattening is not deterministic, representing a problem when the robot skin feedback is exploited in control loops, where hard real-time capabilities are usually required.

Therefore, it is the author's opinion that, beyond the computational feasibility, at least the following two requirements should be guaranteed when designing algorithms providing a 2D representation of the whole robot body:

- *fixed distortions*: the introduced distortions should be the same for contacts occurring on the same robot part and they should not depend on the current robot state or on the contact area;

- *time determinism*: the 2D map representing the current robot state (or an approximation of it) must be generated within a specific time deadline.

Instead of performing an on-line parameterization, the proposed solution consists in creating a tactile map for each link (as described in Chapter 3) and then properly connect them. Figure 6.1 shows an example of a contact distributed between two adjacent links.



(a) Contact occuring between two links of a robot. The blue arrow indicates the rotation axis.

(b) Corresponding sensors placement for the current posture.

(c) Pressure Maps of the two links.

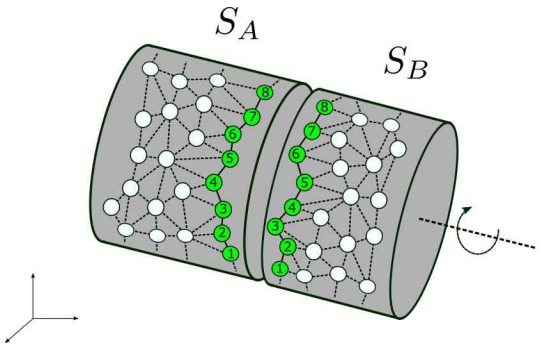
Figure 6.1: The image shows that a contact across two links is split between the two pressure maps. The red markers correspond to tactile elements involved in the contact.

Before creating a tactile image from Figure 6.1(c), the pressure maps need to be connected, possibly preserving the relations across the boundary of the two links.

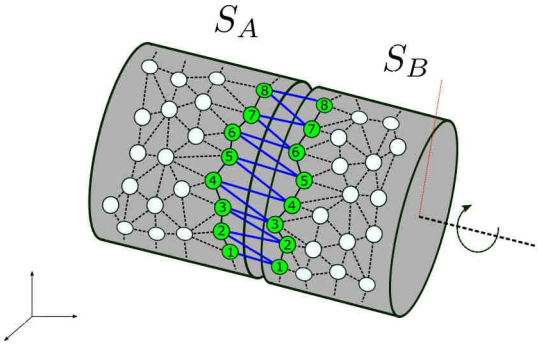
6.2 Proposed Approach

Consider two robot links A and B connected through a rotational joint and covered with robot skin. Figure 6.2(a) schematically describes the links and their associated meshes S_A and S_B. As shown in Figure 6.2(b), for a given joint position, the adjacent taxels on the boundaries of the two links can be *connected* by computing a triangulation. The outcome of this operation is a new set of relations, both *topological* and *geometrical*, represented with blue edges in Figure 6.2(b). The relation is topological since it establishes a **correspondence** between two taxels. It is also geometrical since it defines a **metric** relation between two adjacent taxels. When the posture of the two links changes, the correspondences among the taxels belonging to the boundary of the meshes change accordingly (see Figure 6.2(c)).
The set of adjacent taxels lying on the boundaries can be chosen a priori, but the relations among them must be updated online by recomputing a triangulation.

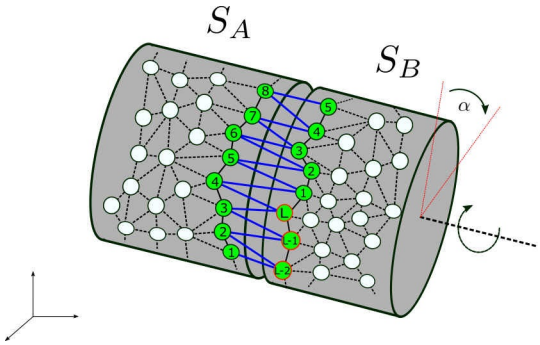
The *topological* correspondences among the taxels computed in 3D can be applied to the tactile maps M_A and M_B as shown in Figure 6.3. The problem arising is then computing a set of geometric relations *approximating* those ones obtained in the 3D space. Actually, M_A and M_B are two distinct tactile maps that can be considered as two *rigid bodies*, as if they were two different sheets of paper. The key idea is computing a suitable set of 2D



(a) Two robot links connected through a rotational joint. S_A and S_B are the two meshes associated to the taxels placement.



(b) The two links connected by computing a triangulation among the adjacent taxels lying at the boundaries.



(c) Supposing to have L taxels on the boundary of the link B, the relations across the adjacent taxels change while the link B is rotated of an angle α.

Figure 6.2: Schematic representation of two robot links covered with robot skin. The taxels lying at the boundaries are labelled with an index. Changing the robot joint position, the triangulation among the adjacent taxels is recomputed.

geometric relations by properly defining a relative displacement between the two maps (a roto-translation) with respect to a reference frame.

6.3 Problem Formulation

The previous idea can be formalized as an optimization problem, where the goal is to find the optimal parameters for performing a roto-translation of M_B that preserves the length of the edges that connect S_A and S_B.

In Figure 6.2, the positions of the taxels that lie on the boundaries are defined as $\mathbf{x}_{\mathbf{A}h} \in \mathbb{R}^3$ and $\mathbf{x}_{\mathbf{B}l} \in \mathbb{R}^3$ for the link A and B respectively, with $h = 1,\ldots,H$ and $l = 1,\ldots,L$. The corresponding positions on M_A and M_B (see Figure 6.3) are defined as $\mathbf{u}_{\mathbf{A}h} \in \mathbb{R}^2$ and $\mathbf{u}_{\mathbf{B}l} \in \mathbb{R}^2$. The set of *topological* connections in Figure 6.2(b), computed for the current posture, is

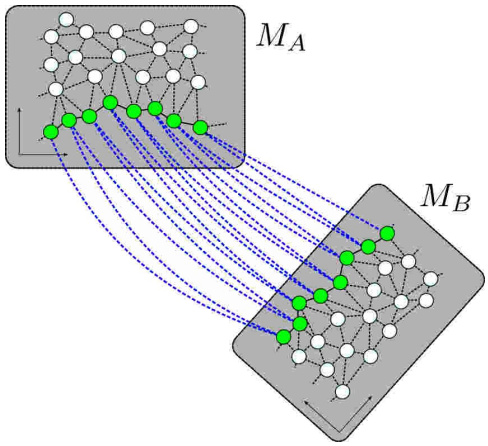


Figure 6.3: Two distinct tactile maps lying on the same plane with their reference frames.

formally described by means of an adjacency matrix $\mathbf{E} \in \mathbb{R}^{H \times L}$:

$$\mathbf{E} = [e_{hl}] = \begin{cases} 1 \text{ iff taxels } h \text{ and } l \text{ are connected} \\ 0 \text{ otherwhise} \end{cases} \tag{6.1}$$

The distances between any pair of taxels lying at the boundaries of interest are expressed using the following matrix:

$$\mathbf{D^x} = [d^x_{hl}] = |\mathbf{x_{A}}_h - \mathbf{x_{B}}_l| \tag{6.2}$$

where $h = 1, \ldots, H$, $l = 1, \ldots, L$ and $\mathbf{D^x} \in \mathbb{R}^{H \times L}$.
The roto-translation operation that must be applied to M_B is defined using a rotation matrix and a translational term:

$$\mathbf{R}(\boldsymbol{\theta}) = \begin{bmatrix} cos(\theta_\alpha) & -sin(\theta_\alpha) \\ sin(\theta_\alpha) & cos(\theta_\alpha) \end{bmatrix}$$
$$\Delta\mathbf{u}(\boldsymbol{\theta}) = [\theta_x \ \theta_y]^T \tag{6.3}$$

where $\boldsymbol{\theta} = [\theta_\alpha, \ \theta_x, \ \theta_y]^T$ is the vector that contains the parameters of the transformation.
In the same way the distances between the taxels in the 2D representation can be defined by means of the matrix $\mathbf{D^u}(\boldsymbol{\theta}) \in \mathbb{R}^{H \times L}$. In this case the transformation applied to the points belonging to M_B must be taken into account:

$$\mathbf{D^u}(\boldsymbol{\theta}) = [d^u_{hl}(\boldsymbol{\theta})] = |\mathbf{u_{A}}_h - [\mathbf{R}(\boldsymbol{\theta})\mathbf{u_{B}}_l + \Delta\mathbf{u}(\boldsymbol{\theta})]| \tag{6.4}$$

Finally, the distances between the corresponding connected components (see Figure 6.2 and 6.3) can be expressed using the following two vectors:

$$\begin{aligned} \mathbf{d^x} &= vec\,(\mathbf{D^x} \circ \mathbf{E}) \in \mathbb{R}^{HL} \\ \mathbf{d^u}(\boldsymbol{\theta}) &= vec\,(\mathbf{D^u}(\boldsymbol{\theta}) \circ \mathbf{E}) \in \mathbb{R}^{HL} \end{aligned} \tag{6.5}$$

where $\circ$ denotes the Hadamard element-wise multiplication between the matrices (Horn and Johnson, 1985) and $vec(\cdot)$ represents the vectorization operator, which basically reshapes the resulting matrix into a column vector.
In the previous equation, $\mathbf{d^x}$ and $\mathbf{d^u}(\boldsymbol{\theta})$ are two sparse vectors (i.e. vectors with many elements identically null) whose components different from zero represent the distances between two *connected* adjacent taxels. Since the goal is to preserve the geometric relations between the 3D and 2D spaces, the vector $\boldsymbol{\theta}$ can be chosen to minimize the difference between $\mathbf{d^x}$ and $\mathbf{d^u}(\boldsymbol{\theta})$ in the sense of the least squares.
Therefore, the optimal choice of the parameters can be found solving the following optimization problem:

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \quad \frac{1}{2}\left[\mathbf{d^x} - \mathbf{d^u}(\boldsymbol{\theta})\right]^T \left[\mathbf{d^x} - \mathbf{d^u}(\boldsymbol{\theta})\right] \tag{6.6}$$

It must be noted that in the equation above the identically null components of $\mathbf{d^x}$ and $\mathbf{d^u}(\boldsymbol{\theta})$ do not affect the cost function. Equation (6.6) is non-linear and its minimization requires a numerical approach. In this work, a Quasi-Newton algorithm (Nocedal and Wright, 2006) has been applied to find a locally optimal solution. The choice of a proper initial state, required by the algorithm, will be discussed in the next Section.

It is worth noting that the proposed approach satisfies the properties discussed in Section 6.1. Indeed, the flattening of the robot links is computed once, thus the introduced distortions are not dependent on the current robot state. Moreover, since Equation 6.6 is solved with an iterative approach, the sub-optimal solutions found at each step are ready to be used. Therefore, even if the optimal solution cannot be found without violating a time deadline, an approximate solution is always available.

6.4 Experimental Results

The approach has been tested on a Baxter robot partially covered with robot skin (see Figure 6.4). In this experiment, only the upper part of the links has been considered, which hosts 1480 taxels. Further details about the skin technology and the sensorized robot arm can be found in Appendices A and B.

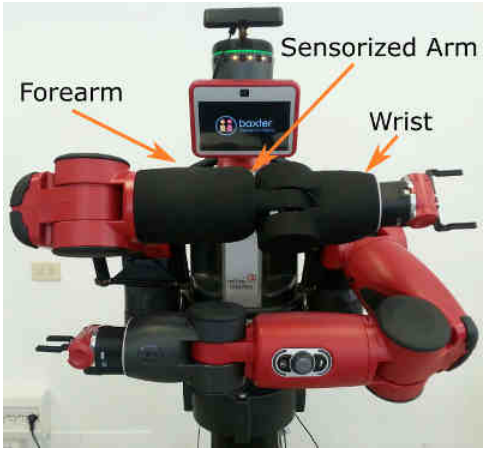


Figure 6.4: Baxter robot with a sensorized arm.

Figure 6.5 shows the corresponding tactile maps associated to the robot links. The taxels colored in green represent the adjacent elements lying at the boundaries of interest and used to build relations while the joint is moving.

The particular geometry of the Baxter links allows to simplify the cost function in Equation (6.6). To clarify this point, consider Figure 6.6, where the forearm and the wrist of the Baxter robot are shown.

The two tactile maps can be pre-aligned (as described in the image) since a rotation of the wrist in the 3D space always corresponds to a translation of its tactile map in the 2D space. Indeed, rotating the joint produces a translational movement in the horizontal axis, as indicated by the arrows in Figure 6.6.

In this condition, the optimal rotation angle can be pre-computed, solving only a problem of translation and removing the strong non-linearities in Equation (6.6) introduced by the trigonometric functions. This assumption does not hold for a generic joint and in that case the general problem must be solved.

Another consideration concerns the initial solution $\boldsymbol{\theta_0}$ required as a starting point by the Quasi-Newton algorithm (Nocedal and Wright, 2006). A good choice of it reduces the number of steps taken by the solver. Since the movement of any type of robot joint is limited to a fixed range, the movement of the tactile maps is limited too.

For these reasons, an initial solution inside the range of movement of the two maps is considered. In particular, we chose the midpoint of this range as a starting point for the minimization process. The pre-alignment of the tactile maps and the choice of $\boldsymbol{\theta_0}$ allow the optimization algorithm to converge after only three iterations.

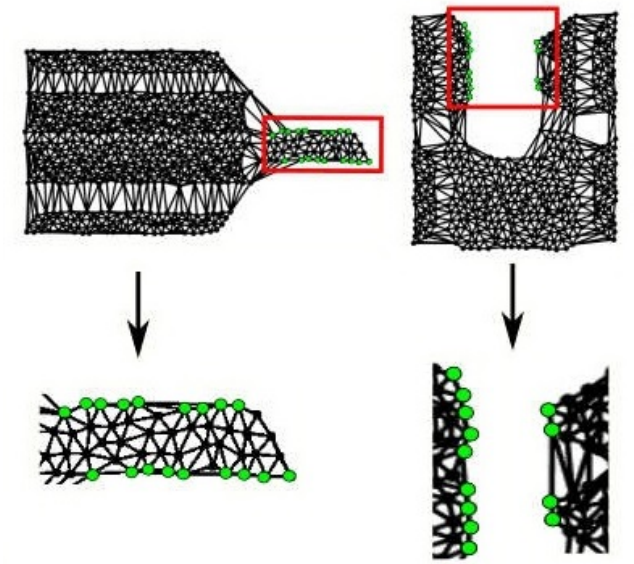


Figure 6.5: Green points represent the selected adjacent taxels used to compute the online triangulation in the 3D space.

Contacts performed with a hand have been applied between the forearm and the wrist of the Baxter robot in three different joint postures: one with the links aligned, and the other two with the joint close to its upper and lower limits. The choice of these three robot configurations is motivated by the fact that they fully describe the movements of the pressure maps for these robot links.

The taxel positions and their pressure measurements have been recorded in each configuration of the Baxter arm. The approach explained in the previous Section has been implemented in Matlab and tested using the collected data.

Figure 6.7 shows three different images which summarize the experiments. Each subfigure contains:

1. A picture of the contact.

2. The 3D position of the taxels for the current posture. In this image, the taxels involved in the contact are colored in red, while the relations established between the adjacent taxels are marked with blue edges. The area containing the relations is zoomed in and shown in a different perspective for sake of clarity.

3. The output of the proposed method, that moves the pressure maps preserving the geometric relations.

4. The resulting tactile image obtained by resampling the pressure map.

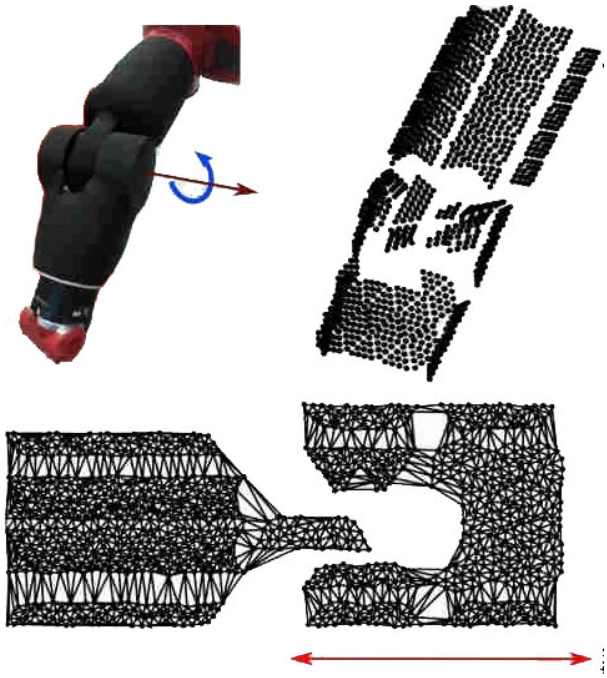


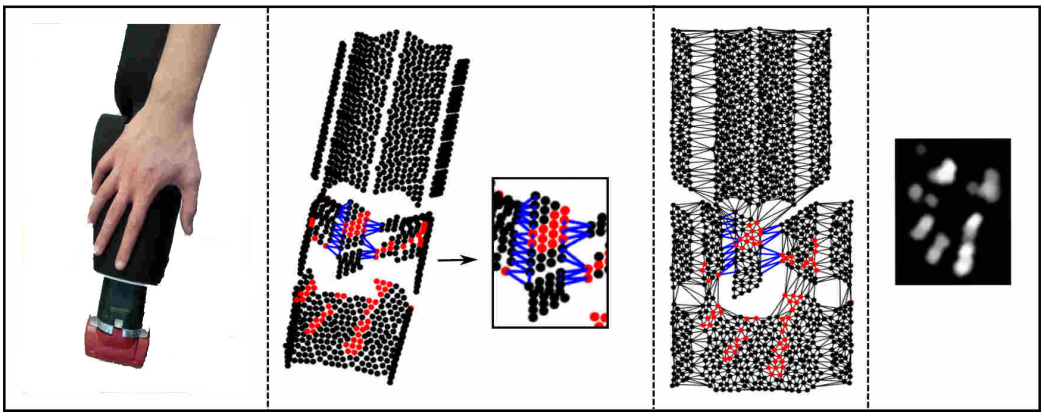
Figure 6.6: Tactile maps pre-aligned. The red arrow shows the possible movements of the tactile map representing the Baxter wrist.

The experiments in Figure 6.7, show that the topological and geometrical relations across the boundary change depending on the robot posture. When the links are moving, due to their geometry, some of the edges in 3D are stretched too much and in this case we removed the edges whose length is greater than a threshold value of 3 *cm*.

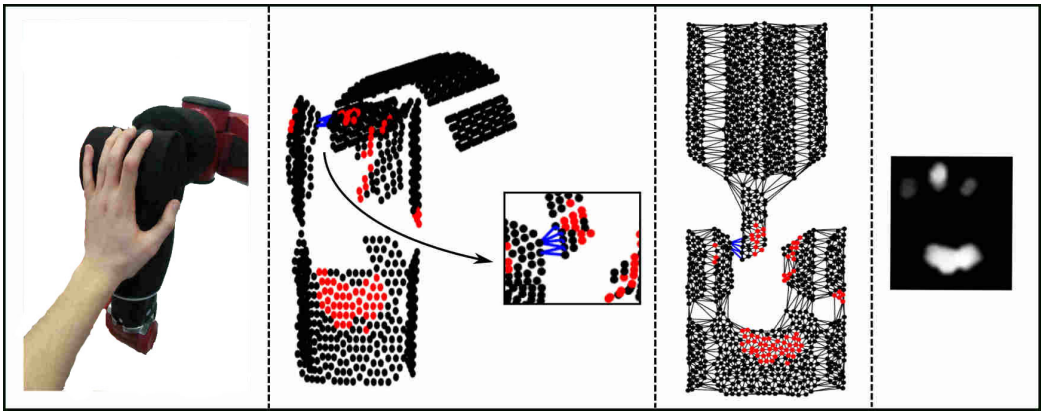
Figure 6.7(a) shows the configuration of the pressure maps when the robot arm is distended. The maps have been correctly aligned and the resulting tactile image preserves the contact shape.

In Figure 6.7(b) a contact is applied when the joint is close to its lower bound. In this experiment, the edges that interconnect the adjacent taxels are removed from one side of the links since their length exceed the given threshold value. It can be seen that in this configuration the tactile maps are moving away in order to correctly preserve the geometric relations. The more the joint is close to its lower limit, the more the maps will move away from each other.

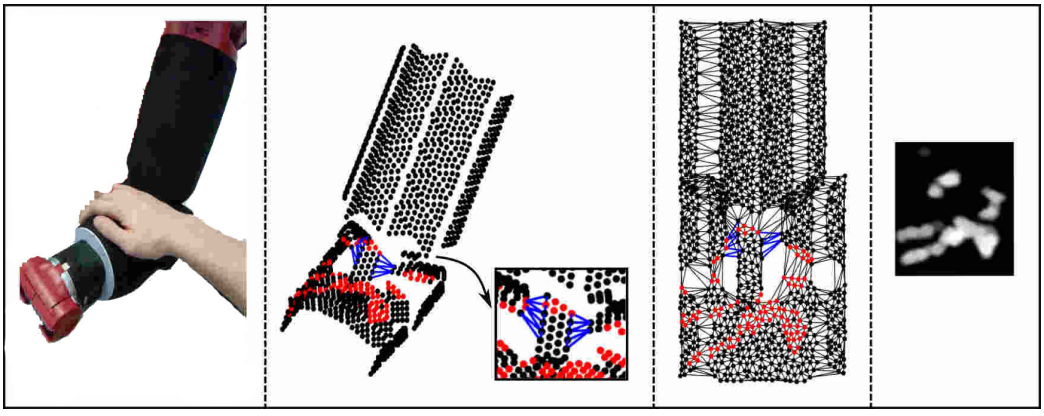
Conversely, Figure 6.7(c) shows the opposite situation, where the joint is close to its upper bound. While the joint is reaching its limit, the pressure maps get closer to each other. In this particular configuration, the two tactile maps need to be overlapped in order to preserve the distances. The three experiments show that in all the presented cases, the contact shape and the resulting tactile image are correctly reconstructed.



(a) Forearm and wrist aligned



(b) Joint near to its lower bound



(c) Joint near to its upper bound

Figure 6.7: Experiments in three different joint postures. From left to right: (i) picture of the contact; (ii) taxels placement, including a detailed information about the interconnections of adjacent taxels belonging to different links; (iii) the interconnected pressure maps; (iv) the resulting tactile image.

6.5 Conclusion and Discussion

The problem of generating a tactile map of the robot body has been introduced and preliminarily addressed in this Chapter.
As previously said, the contents of this Chapter are not conclusive. Although the problem has been formalized in a general view, only the case with two links partially sensorized has been tested. However, the presented results can be seen as a starting point for a further generalization of the method. Indeed, several aspects can be improved proposing additional analysis or experiments. For example, different types of joints (also with different rotation axes from the one shown in the experiments) and robot geometries should be considered. Then the method has to be generalized considering an arbitrary number of links.

Chapter 7

Robot Skin Spatial Calibration

7.1 Introduction

The problem of the robot skin spatial calibration has been introduced in Section 2.3. Within the scope of this work, the calibration on the platform described in Appendix B has been performed manually. Although the manual calibration can be imprecise or approximative, the level of accuracy required should be weighted considering the particular operating context. In this work a reasonable accurate spatial calibration (in relation to the proposed applications) has been possible since the robot links have a fairly simple geometry. However, when the robot geometry is more complex or a high level of accuracy is needed, a manual calibration could be not enough. As an example, consider the work proposed by Roncone et al. (2014). A self-touch strategy has been implemented to calibrate the kinematic chain of the iCub robot. Tactile feedback is used to retrieve the contact location which is considered as a ground truth to optimize the robot kinematic model. In such context, a rough spatial skin calibration could negatively impact the overall performance of the approach.

The works discussed in Section 2.3 are based on an a priori knowledge of the tactile system or the robot body, executed in a structured environment or tailored to a specific robot skin technology. This Chapter introduces a novel approach aimed at removing those dependencies, allowing to: (i) allow the robot to perform the calibration autonomously and to learn its shape by tactile exploration, (ii) extend its applicability to different robotic skin technologies.

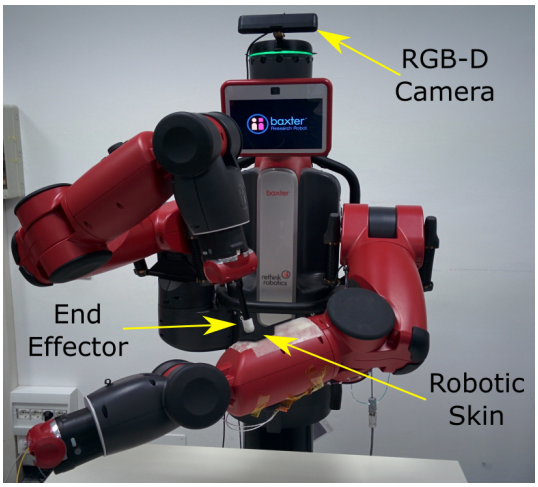


Figure 7.1: The experimental setup used in the robotic skin spatial calibration experiments. The dual arm Baxter robot is equipped with a RGB-D camera and a robotic skin partially covering one of the forearms.

7.2 Proposed approach

The method proposed in this Chapter assumes that the robot is covered with a robot skin and is *capable of activating the sensors* (see Figure 7.1). Moreover, the robot is equipped with a vision system that can view the body area involved in the calibration procedure and *can reconstruct its shape*. The minimum requirements for the applicability of the method are highlighted in the following:

- the robot knows its proprioceptive information including joint torques and its kinematic structure;
- the robot is equipped with RGB-D cameras or equivalent sensors that can view the area of the robot skin subject of the calibration procedure (see Figure 7.1);
- the robot has one arm that can be used for exciting the robot skin sensors subject of the calibration procedure (see Figure 7.1).

The approach has been tested using the robot skin described in Appendix A. However, the same techniques can be applied to other kinds of transducers as long as *the sensor response to the same stimulus depends solely on the relative displacement between its application point and the sensor position*. The steps of the calibration procedures can be summarized as follows. Firstly, the robot recognizes its body within its field of view correlating proprioceptive information with the scene captured by the vision system. Secondly, the robot individuates

an area where to start the calibration procedure and determines the best way for approaching the robot skin accordingly to its shape. Thirdly, as soon as the contact is detected from skin measurements, the robot controls the exerted force in order to produce a controlled stimulus. The contact centroid, inferred from the robot pose, is recorded and the taxels responses are associated with it. Finally, after collecting this information, the robot releases the contact and repeats the same procedure in another place on the skin. After collecting the samples, an algorithm infers the position of each sensor with respect to a reference frame of the robot body, correlating the sensors responses with the position of the related stimulus. In this Section, the parts necessary for implementing the procedure are described in detail.

7.2.1 Robot body detection

In the first step of the calibration procedure, the robot has to determine a location on its body where to exert the force for activating the sensors. However, the lack of knowledge on the model of the robot body does not allow a direct planning of the end-effector motion. The robot has to reconstruct the shape of its body using the knowledge of its kinematic structure and the measurements coming from an RGB-D sensor looking at the body part subject of the calibration procedure. The problem is tackled as follows. Firstly, an area on the RGB-D image containing the robot body part is segmented using a projection of the kinematic structure of the robot in the image (see Figure 7.2) and an estimate of the robot body overall dimensions (see Figure 7.3 and 7.4). Secondly, depth information is used for separating the robot body from the background.

Suppose that the kinematic structure of the robot is known and can be described by kinematic chains originating from the robot base b and composed by N joints $i = 1, 2 \ldots, N$, by links $\mathrm{L}_{i,j}$ connecting two consecutive joints i - j and by links $\mathrm{L}_{i,b}$ connecting joint i to the robot base b. Moreover, consider $<i>$ as the reference frame of joint i, $<b>$ as the robot base reference frame and $<c>$ as the RGB-D sensor reference frame fixed with respect to one of the joint reference frames. From the kinematic structure, it is possible to compute the transformation matrices ${}^{b}\mathbf{T}_{i}$ relating each joint reference frame $<i>$ to $<b>$, as well as the transformation ${}^{b}\mathbf{T}_{c}$ relating the RGB-D sensor reference frame $<c>$ to $<b>$. By defining ${}^{b}\mathbf{o}_{i}$ as the position of joint i with respect to the base $<b>$ in homogeneous coordinates, the computation of the position of each joint with respect to the camera frame can be expressed as ${}^{c}\mathbf{o}_{i} = {}^{c}\mathbf{T}_{b}\,{}^{b}\mathbf{o}_{i}$ and consequently, their projection in the camera image plane are obtained as follows (Corke, 2013):

$$\widetilde{\mathbf{p}}_{i} = \begin{bmatrix} \widetilde{u}_{i} \\ \widetilde{v}_{i} \\ \widetilde{w}_{i} \end{bmatrix} = \begin{bmatrix} k_{x} & 0 & u_{0} & 0 \\ 0 & k_{y} & v_{0} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} {}^{c}\mathbf{o}_{i} \tag{7.1}$$

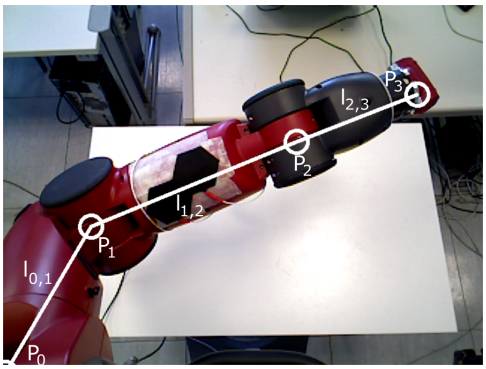


Figure 7.2: The projection of joints position and links in the image.

where $\widetilde{\mathbf{p}}_i$ is the position of joint i in the picture in homogeneous coordinates, u_0 and v_0 are the coordinates of the intersection point between the optical axis and the image plane, k_x and k_y are ratios between the size of the pixel along x and y axes of the image plane and the focal length. From 7.1, the non-homogeneous coordinates can be computed as:

$$\mathbf{p}_i = \begin{bmatrix} u_i \\ v_i \end{bmatrix}, \; u_i = \frac{\widetilde{u}_i}{\widetilde{w}_i}, \; v_i = \frac{\widetilde{v}_i}{\widetilde{w}_i}$$

The result is reported in Figure 7.2, where it can be seen that the visible joints positions are highlighted using circles. The coordinates $\mathbf{p}_i$, as well as the segments $\mathrm{l}_{i,j}$, connecting $\mathbf{p}_i$ with $\mathbf{p}_j$, build up the skeleton of the robot arm (see Figure 7.2 and 7.4) and are the starting point for defining a containing polygon of the arm in the image. Each pixel of $\mathrm{l}_{i,j}$ is projected back in 3D space using the related depth information (Corke, 2013) and the distance between each projected point and the robot link $\mathrm{L}_{i,j}$ is computed. The maximum, defined as $d_{max_{i,j}}$, is selected as a rough estimate of the depth of the robot body surrounding the link (see Figure 7.3) and, finally, the equivalent distance $\bar{d}_{max_{i,j}}$ in pixel is computed. This information is used to draw a polygon in the image that contains the robot body (see Figure 7.4).

The polygon is constructed as follows. For each $\mathrm{l}_{i,j}$, two lines parallel to the link and distant $\bar{d}_{max_{i,j}}$ on each side are drawn (see Figure 7.4). The polygon is constructed by finding the intersection points of each line with the lines of the previous and next links. The polygon is closed by two lines perpendicular to the first and the last link and passing through the first and the last joint respectively, as in Figure 7.4. Finally, considering the part of the image contained by the polygon, each pixel is marked as part of the robot body if the related depth is less than the depth of the closest point on a link $\mathrm{l}_{i,j}$ plus the estimated link depth, i.e. the related $d_{max_{i,j}}$. The outcome is a binary mask that individuates the pixels in the image that are part of the robot body. The mask is applied to the original image in order to segment

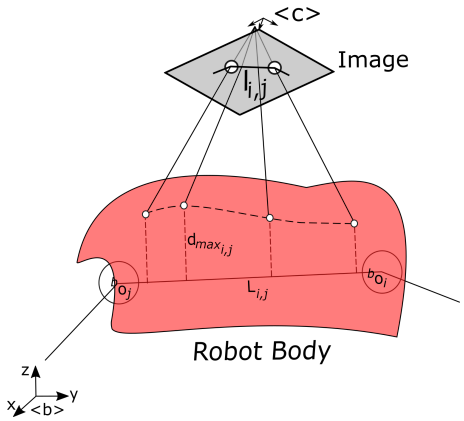


Figure 7.3: The projection of the pixels of a link on the robot body.

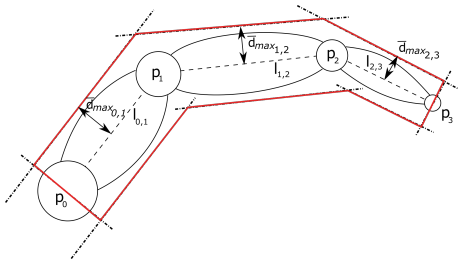


Figure 7.4: A sketch of the robot arm as seen from the camera and, in red, the polygon containing the robot arm.

the robot body from the background, as can be seen in Figure 7.5. It must be noted that the presented approach does not depend on any parameter and is suited for an autonomous calibration performed by the robot.

7.2.2 Approaching the robot body

Once the robot body part is segmented from the rest of the RGB-D image, the robot has to select a point where to execute the calibration procedure. Supposing that the robot is fully covered with a robot skin, any point on the body can be selected for performing the calibration. In principle, at this step, a random location can be picked by the robot. Nonetheless, it is evident that the selected area could be not reachable (e.g., the back of the robot) or difficult to touch with a stable contact (e.g., a border). While the former requires the robot to rely on external elements in a known pose in order to activate the sensors, for the latter the robot can change its posture to facilitate the task. In both the cases, the presented approach still applies. In this work, it is assumed that the area subject of calibration can be reached and any point in

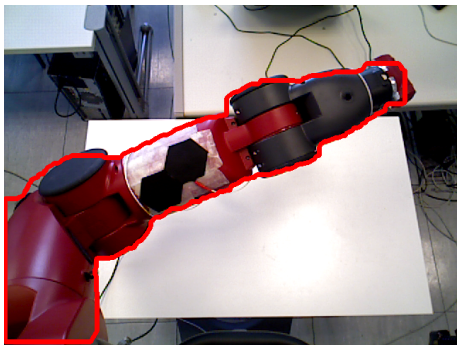


Figure 7.5: The contour of the area selected for the calibration is superimposed on the original image.

it can be randomly selected by the robot. Once a point in the area individuated in the previous step is selected, it can be projected on the robot body using the depth measurements from the camera. The outcome is a target position $\mathbf{x_t} \in \mathbb{R}^{3\times1}$ defined with respect to a robot reference frame. While in contact, the robot has to apply a force in a controlled way avoiding slippage and guaranteeing that the stimulus can be repeated and is uniform around the application point, i.e. the robot has to exert a force along the normal on the robot body at the desired point of contact. As reported in Holz et al. (2012), the normal to the surface can be computed from the point cloud measured by the depth sensor of the camera. The point cloud around $\mathbf{x_t}$ is considered and the normal is computed. The result is the vector $\mathbf{n_t} \in \mathbb{R}^{3\times1}$.

7.2.3 Robot control

In the following, it is described how the robot approaches the target position and exerts a controlled force required for activating the skin sensors. The procedure can be described by the following steps:

- the robot is controlled to move the end-effector, that in our case is a finger-like probe (see Figure 7.1), over the target position $\mathbf{x_t}$ and to align it with the normal $\mathbf{n_t}$;
- then the end-effector is moved towards $\mathbf{x_t}$ along $\mathbf{n_t}$ until the contact is detected by the underlying skin sensors, as can be seen in Figure 7.1;
- finally, the robot is controlled for maintaining the contact position $\mathbf{x_t}$ while applying a constant force along $\mathbf{n_t}$.

In the former part, it is required to control the position and the orientation of the end-effector until a contact is reached at the target position. In the latter, the force along $\mathbf{n_t}$ must be

controlled together with the position on the plane orthogonal to $\mathbf{n_t}$ in order to keep the end-effector in $\mathbf{x_t}$ aligned with $\mathbf{n_t}$. In view of this, the most suitable control law for this task is the hybrid position/force robot control. The controller has been implemented according to what is reported in Fisher and Mujtaba (1992). Finally, considering that the end-effector is a finger-like probe commanded to exert a force along $\mathbf{n_t}$, its tip coincides with the contact centroid. Therefore, the contact centroids can be estimated by computing the end-effector position.

7.2.4 Tactile sensors pose estimation

When the robot reaches a stable contact condition, the computed position of the contact centroid is associated to the measurements of the tactile sensors. The procedure is repeated N times on the robot body to build up the dataset composed of contact centroids $\mathbf{x}_i \in \mathbb{R}^{3\times 1}, i \in \{1,2,\ldots,N\}$ and the vectors $\mathbf{t}_i \in \mathbb{R}^{M\times 1}$ containing the responses of the M taxels, of the robot skin associated with the contact i. The characteristic response of each sensor is different from the others and it is not possible to directly compare the measurements of two different sensors without an a priori force calibration. However, it can be assumed that the response of a single sensor is repeatable thus the raw samples can be directly processed without a previous measurement calibration. For this reason, the robot skin spatial calibration procedure is divided in *a set of independent estimation problems, each one related to the pose estimation of a single taxel j on the robot body*. It should be clear that, in this way, the solution of each estimation problem is *totally independent* from the *responses of the other taxels, the arrangement, spatial resolution and shape and solely depends on the measurements of the sensor*.

The dataset of each problem j is defined as:

$$T_j = \left\{\mathbf{x}_i, t_{i,j}\right\} \forall i \in \{1,2,\ldots,N\} : t_{i,j} \neq 0,$$

where $t_{i,j}$ denotes element j in vector $\mathbf{t}_i$, i.e. the measurement of taxel j for contact i. For sake of clarity, the pose estimation problem is discussed considering the single taxel, implying that the same approach can be used for each taxel of the robot skin. Thus, the dataset is defined as:

$$T = \{\mathbf{x}_k, t_k\} \forall k \in \{1,2,\ldots,L\}$$

that is the set of L contact centroids $\mathbf{x}_k$ and the related responses t_k of the sensor.

Considering that the force stimulus is filtered by the mechanical low-pass filtering effect introduced by the layers composing a robot skin (Shimojo, 1997) and thanks to the

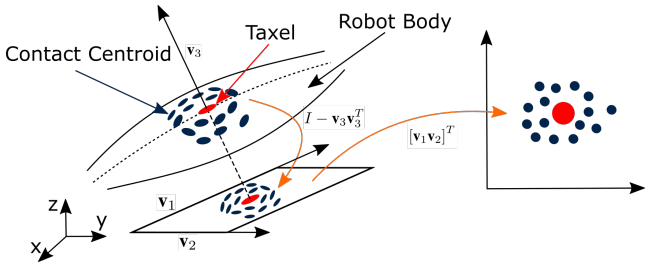


Figure 7.6: Contact centroids projection

experimental results reported in Maiolino et al. (2013), it is reasonable to assume that the response of a taxel to the same stimulus solely depends on its application point and can be approximated with a Gaussian function around the sensor center. Therefore, we can see the taxel pose estimation problem as the *estimation of the center of the Gaussian function best fitting the sensor response* as long as the following assumptions are met:

- each sensor provides a local measurement on the applied stimulus and can be considered as a punctual sensor;
- the end-effector can be used to generate contact areas with a well-defined contact centroid and comparable to the spatial resolution of the robot skin (i.e. we are excluding multi-point or large area contacts).

These assumptions allow to neglect the curvature of the surface around the taxel and, consequently, it is possible to simplify the problem to a *bi-dimensional Gaussian function estimation*.

The estimation problem is tackeld as follows. Given the set of $\mathbf{x}_k$, it is possible to compute the normal to the plane individuated by them (Holz et al., 2012). As a result, we can build an orthonormal basis of $\mathbb{R}^3$ where one of the unit vectors, i.e. $\mathbf{v}_3$, is the computed normal and the other two, $\mathbf{v}_1$ and $\mathbf{v}_2$, are any two orthonormal vectors parallel to the plane. The result is a new reference frame centred with the origin where it is possible to project the points $\mathbf{x}_k$ using the following relation:

$$\hat{\mathbf{x}}_k = [\mathbf{v}_1\mathbf{v}_2]^T \left(I - \mathbf{v}_3\mathbf{v}_3^T\right) \mathbf{x}_k$$

with $\hat{\mathbf{x}}_k \in \mathbb{R}^2$, that is a composition of a projection of the points on the plane individuated by $\mathbf{v}_1$ and $\mathbf{v}_2$ and a rotation of the points on the reference frame described by the orthonormal basis (see Figure 7.6).

Coherently with the hypothesis previously discussed, the taxel response can be modelled with the following function:

$$\hat{t}(\hat{\mathbf{x}}) = \beta_5 \exp\left(-\left(\frac{(\hat{x}_1-\beta_1)^2}{\beta_3^2}+\frac{(\hat{x}_2-\beta_2)^2}{\beta_4^2}\right)\right), \tag{7.2}$$

which represents a bi-dimensional Gaussian model where $\hat{x}_1$ and $\hat{x}_2$ are the components of a contact centroid $\hat{\mathbf{x}}$ and $\beta_1, \beta_2, \beta_3, \beta_4$, and β_5 are the Gaussian parameters that can be conveniently represented in vectorial form $\boldsymbol{\beta} = [\beta_1, \beta_2, \beta_3, \beta_4, \beta_5]^T$.

Supposing to have L elements in the dataset T, the goal is to fit this data with the Gaussian model described in (7.2) by minimizing the mean square error defined as:

$$J(\boldsymbol{\beta}) = \frac{1}{L}\sum_{k=1}^{L}[t_k - \hat{t}_k(\hat{\mathbf{x}}_k, \boldsymbol{\beta})]^2 \tag{7.3}$$

According to the hypothesis, the estimated position of the taxel will be given by parameters β_1 and β_2, which represent the center of the Gaussian model. It is now possible to define the tactile sensor position estimation as:

$$\boldsymbol{\beta}^* = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \quad J(\boldsymbol{\beta})$$

where $\boldsymbol{\beta}^*$ is the vector that minimizes Equation (7.3). The cost function (7.3) is non linear and requires a numerical approach for finding the solution. In order to solve the minimization problem, the Newton-Raphson algorithm (Nocedal and Wright, 2006) is exploited to iteratively seek the solution according to the following equation:

$$\boldsymbol{\beta}_{l+1} = \boldsymbol{\beta}_l - \mathbf{H}(\boldsymbol{\beta}_l)^{-1}\nabla J(\boldsymbol{\beta}_l) \tag{7.4}$$

where $\boldsymbol{\beta}_l$ is the solution at step l, $\nabla J(\boldsymbol{\beta}_l) \in \mathbb{R}^{4x1}$ is the gradient of J evaluated in $\boldsymbol{\beta}_l$ and $\mathbf{H}(\boldsymbol{\beta}_l) \in \mathbb{R}^{4x4}$ is the Hessian matrix of J evaluated in $\boldsymbol{\beta}_l$. It must be noted that if $\mathbf{H} \leq 0$, equation (7.4) cannot be used for moving towards the solution (Arora, 2015). In this case, the gradient descent method can be used:

$$\boldsymbol{\beta}_{l+1} = \boldsymbol{\beta}_l - \alpha\nabla J(\boldsymbol{\beta}_l)$$

where α is a positive number. The initial state $\boldsymbol{\beta}_0$ of the algorithm is set as:

$$\boldsymbol{\beta}_0 = \begin{bmatrix} \bar{\mathbf{x}} \\ \sigma_{max} \\ t_{max} \end{bmatrix}$$

where $\bar{\mathbf{x}} \in \mathbb{R}^2$ and $\sigma_{max} \in \mathbb{R}^2$ are respectively the mean and the maximum standard deviation of $\hat{\mathbf{x}}_k$ and t_{max} is the maximum of the sensors responses t_k. The algorithm iterates until the following criterion is met:

$$||\boldsymbol{\beta}_l - \boldsymbol{\beta}_{l-1}|| < \varepsilon$$

where $||.||$ is the euclidean norm and ε is a threshold on the minimum update distance.

Once the solution $\boldsymbol{\beta}^*$ is obtained, the first two elements are the sensor position estimation with respect to the previously defined reference frame. The estimation of the taxel position $\tilde{\mathbf{x}}$ with respect to the robot reference frame can be obtained as

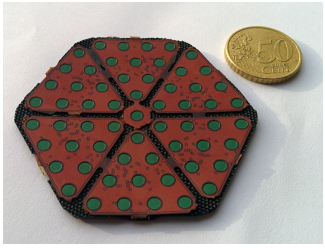
$$\tilde{\mathbf{x}} = [\mathbf{v}_1\mathbf{v}_2\mathbf{v}_3][\beta_1^*, \beta_2^*, 0]^T + \mathbf{v}_3\mathbf{v}_3^T\mathbf{x}^*$$

where $\mathbf{x}^* = \frac{1}{L}\sum_{k=1}^{L}\mathbf{x_k}$. Finally, it should be noted that the estimate of the orientation of the sensor can be limited to the estimate of its normal, that is $\mathbf{v}_3$.

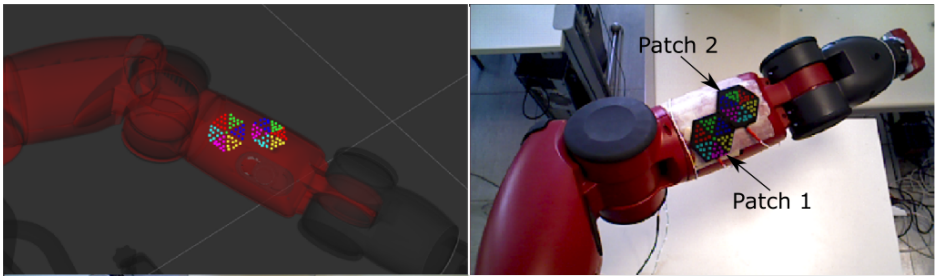
7.3 Experimental Results

The experiments have been performed on a Baxter robot with the addition of an Asus Xtion Pro Live RGB-D sensor mounted on the head in a known pose. The camera is oriented in a way that the robot can view the area in front of it (see Figure 7.1 and 7.5). The end-effector of the robot is made by a finger mounted on the robot gripper (see Figure 7.1) and its tip is a semi-sphere of radius 1 cm. The robot has to calibrate a number of taxels organized as hexagonal patches (see Figure 7.7(a)). Two hexagonal patches cover the forearm of the robot as reported in Figure 7.7(b).

The proposed approach has been evaluated in two different types of experiments. In the former, the robot has to calibrate a hexagonal patch placed on a table in a *known position and orientation*. In this experiment, it is possible to evaluate the estimation accuracy by comparing the obtained results with the ground truth and thus to evaluate the absolute (ABS) spatial calibration of the sensors. In the latter, the robot has to calibrate the two hexagonal patches attached to the forearm. In this scenario, the real positions of each sensor is not known and the results will be evaluated according to a relative displacement with respect



(a) A hexagonal patch of the robot skin.



(b) The result of the spatial calibration performed on the two hexagonal patches. On the left, the sensors are reported on the robot model. On the right, the sensors are projected on the image captured by the RGB-D camera.

Figure 7.7: The hexagonal patch and results of the spatial calibration.

to a planar model of the two patches. In this case, the results express the accuracy of the relative (REL) spatial calibration of the sensors. Both the experiments followed the procedure described in Section 7.2, summarized as follows:

1. the robot looks at the scene and detects the area where to start the calibration;
2. it selects a location where to apply the force with the end-effector;
3. estimates the normal at the contact position using the depth image coming from the RGB-D sensor;
4. it moves the end-effector to the contact position controlling the exerted force at the desired intensity and with a direction aligned with the previously determined normal;
5. it acquires the measurements from the sensors;
6. it moves the end-effector away from the area of interest;
7. the procedure starts over again at 2 until the samples set is built;

Table 7.1: Experimental results.

Experiment	$\bar{e}$	σ	$\bar{e}_\theta$	e_{max}
Known Pose Patch (ABS)	1.9 mm	0.8 mm	0.06 rad	4.0 mm
Robot Patch 1 (REL)	2.0 mm	1.1 mm	-	6.5 mm
Robot Patch 2 (REL)	2.9 mm	1.6 mm	-	6.8 mm

During the experiments, a force of 5 N has been applied in all the contacts. The force exerted by the end-effector is estimated by the joint torque measurements after compensating for the gravity load.

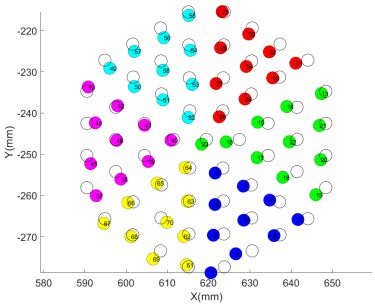
In the first experiment, the robot executes the calibration procedure on a hexagonal patch, consisting of 61 pressure sensors, placed on a table in front of it in a known pose. In this experiment, phase 1 and 2 of the calibration procedure are driven by a user and the rest is repeated 350 times to build up the dataset. Finally, the algorithm described in 7.2.4 is applied for all the 61 sensors and their pose is estimated. The performance of the algorithm has been evaluated on the position and orientation errors. In the former, the mean, the standard deviation and the maximum position error over all the sensors have been computed as:

$$
\begin{aligned}
e_j &= \sqrt{\left(x_j - \tilde{x}_j\right)^2 + \left(y_j - \tilde{y}_j\right)^2 + \left(z_j - \tilde{z}_j\right)^2} \\
\bar{e} &= \frac{1}{M}\sum_{j=1}^{M} e_j \qquad (7.5) \\
\sigma &= \sqrt{\frac{1}{M-1}\sum_{j=1}^{M}(e_j - \bar{e})^2} \qquad (7.6) \\
e_{max} &= \max_j e_j \qquad (7.7)
\end{aligned}
$$

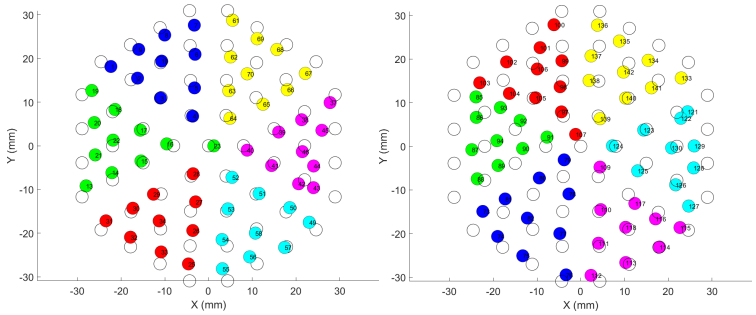
where M is the number of sensors, x_j, y_j and z_j are the real coordinates of sensor j, while $\tilde{x}_j$, $\tilde{y}_j$ and $\tilde{z}_j$ are the related estimates. For the latter, the orientation error can be defined as:

$$
\bar{e}_\theta = \frac{1}{M}\sum_{j=1}^{M} \cos^{-1}\left(\tilde{\boldsymbol{\theta}}_j \cdot \mathbf{n}\right) \qquad (7.8)
$$

that is the average angle between the normal to the plane $\mathbf{n}$ and the estimated one $\tilde{\boldsymbol{\theta}}_j$ for the taxel j. The numerical results are reported in Table 7.1 and the outcome of the calibration procedure is reported in Figure 7.8(a) where the estimated position of the sensors are superimposed to the planar model. The obtained results are quite remarkable. The



(a) The result of the spatial calibration performed on a hexagonal patch placed on the table in front of the robot.



(b) The result of the spatial calibration of the two patches on the robot forearm is compared to a planar model in the origin. On the left, the hexagonal patch number 1. On the right, the hexagonal patch number 2.

Figure 7.8: Planar representation of the experimental results. The expected positions of each sensor in the planar model are represented by white circles with the same diameter of the real sensors, while the estimated positions are presented with numbered circles with different colors for each triangular module.

position error is comparable to the radius of the pressure sensors (1.75 mm), the normal estimation is less than 4° and the maximum position error stays in the half of the pitch between taxels (8 mm). Moreover, a small standard deviation highlights that the procedure performs quite well in average, as can be qualitatively seen in Figure 7.8(a). In order to evaluate the dependency of the calibration results on the number of samples, the procedure has been applied to an increasing subset of samples for each taxel and the related mean and the standard deviation have been computed according to (7.5) and (7.6). The same procedure has been repeated 10 times considering different samples in each subset. The mean among all the trials is reported in Figure 7.9. As it can be noted, the error quickly converges after obtaining 16 samples for each taxel.

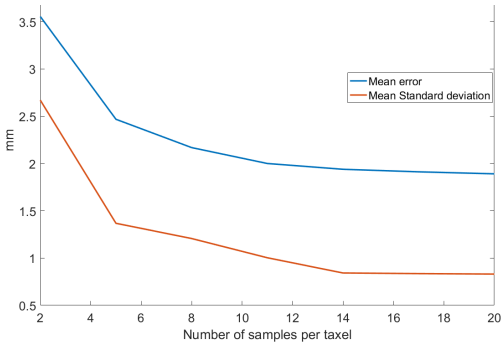


Figure 7.9: Mean $\bar{e}$ and σ computed on an increasing number of samples for each taxel over 10 trials on different datasets.

In the second experiment, the calibration procedure is applied to two hexagonal patches on the forearm of the robot forming a robotic skin of 122 sensors. In this case, the robot is not fully covered with a robotic skin thus it cannot randomly select an area where to perform the calibration. Step 2 of the procedure is not performed autonomously and is driven by an user. The result of this experiment cannot be evaluated using the real pose of the sensors because it is not known in advance. Instead, the spatial calibration result is flattened and compared to a planar model of the two patches by superimposition. In this case, we can compute relative displacement errors between them and the related statistics using equations similar to (7.5), (7.6), (7.7) and (7.8) where the expected pose of each sensor is the pose in the planar model. The results are reported in Table 7.1 and the reconstruction of each patch can be seen in Figure 7.8(b) and Figure 7.7(b), where the estimate of the position of the sensors is reported on both, the model of the robot and an image captured by the RGB-D. Finally, the current approach has been compared to previous works for both absolute and relative errors. In Table 7.2, the directly comparable values have been reported (a dash indicates that the related data is missing in the publication). As can be noted, the proposed approach outperformed the previous ones in all the cases.

7.4 Conclusion and Discussion

This Chapter presents a novel approach to the robotic skin spatial calibration problem based on vision and self-touch. Compared to previous works (Cannata et al., 2010; Del Prete et al., 2011; Mittendorfer et al., 2014), this method requires just a little knowledge on the robot (i.e. its kinematic structure) and a few assumption on the sensors composing the robotic skin, thus can be easily ported to other robots equipped with different skin technologies.

Table 7.2: Comparison with previous works.

First Author	$\bar{e}$	σ	$\bar{e}_\theta$
Cannata et al. (2010) (ABS)	$6.8\,mm$	$10\,mm$	-
Mittendorfer et al. (2014) (ABS)	$32.7\,mm$	-	$0.06\,rad$
Proposed Approach (ABS)	$1.9\,mm$	$0.8\,mm$	$0.06\,rad$
Del Prete et al. (2011) I (REL)	$7.2\,mm$	$3\,mm$	-
Del Prete et al. (2011) II (REL)	$6.6\,mm$	$2.9\,mm$	-
Mittendorfer et al. (2014) (REL)	$\leq 10\,mm$	-	≤ 0.07 rad
Proposed Approach (REL)	$\leq 2.9\,mm$	$1.6\,mm$	-

The calibration of the whole skin system is split in independent estimation problems, each one related to the estimation of the pose of a single taxel. This formulation allows to solve the estimation problems in parallel and while the robot is exploring the tactile system. Finally, the experimental results obtained in the tests performed on the Baxter robot show that the method can effectively reconstruct the sensors arrangement on the robot body. The small estimation errors do not prevent to distinguish one sensor from the others preserving the spatial relations among the taxels composing the robotic skin.

The methods is still not completely autonomous. The presented work highlights the main steps the robot has to perform for the self-calibration of its skin. There are some missing elements that can be added as an extension of the work.

For example, a bi-manual control can be considered. Instead of being passive, the other arm could adjust its position in order to help the robot to touch locations difficult to be reached. Another missing point necessary to achieve an autonomous procedure is a criterion that can be used by the robot for exploring its distributed tactile system.

Part III

Large Area Tactile Sensors for Robot Motion Control

Chapter 8

Robot Control in Unstructured Environments

From Chapters 3 to 6, the robot has been assumed to be passive, indeed no specific robot reactions have been triggered in response to contact events. From this point on, tactile feedback will be used to actively control the robot.

The experiment proposed in this Chapter shows that tactile feedback can be used to guarantee non destructive robot motions when unpredictable contacts or multiple collisions with the environment occur. In particular, the problem of moving a robot arm through obstacles to reach a target position is addressed. It will be shown that robot skin feedback can be used to design a low level tactile-based control law, allowing the robot to safely interact with the environment.

8.1 Introduction and Motivations

The problem of robots moving in presence of obstacles or in cluttered environments has been widely investigated for various classes of robots including: mobile robots (Hoy et al., 2015), robot manipulators (Chitta et al., 2012; Hornung et al., 2012; Khatib, 1985) and snake robots (Sanfilippo et al., 2016). The majority of the proposed approaches requires to know a model of the environment or to build one, typically using vision, and tries to avoid a direct contact with obstacles. Conversely, it has also been demonstrated that, for snake robots locomotion (Sanfilippo et al., 2016), obstacles can, in some cases, be exploited to accomplish the task. The availability of a model of the environment allows to plan in advance the robot actions, but when a model of the environment is not known a priori and vision cannot be used, planning is not possible.

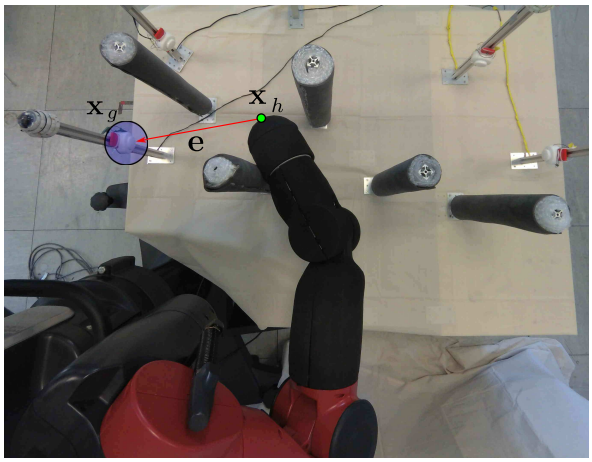


Figure 8.1: A robot is trying to reach a target position while controlling the interaction forces applied on the environment.

A possible approach is to exploit tactile based or other forms of contact force feedback to implement robot control reactive behaviours. The majority of the works addressing this topic uses joint level torque measurements to detect contacts. Examples can be found in De Luca and Flacco (2012), where a robot manipulator is considered, but also in Liljebäck et al. (2014), where contact forces acting on the body of a snake robot are captured from torque sensors. Since collisions can occur while the robot is performing a task, researchers also proposed control techniques that (possibly) preserve the main task execution while the robot is reacting to the contacts (De Luca and Ferrajoli, 2008; Jorda et al., 2019). As previously mentioned in Section 2.2, other works (Calandra et al., 2015; Jain et al., 2013; Killpack et al., 2016) take advantage of robot skin to control the interaction between the robot and the environment.

In this Chapter, the problem of reaching a target point in a cluttered environment is considered. Large area tactile feedback is used in this context to ensure safe and non destructive robot actions. The tactile feedback and the robot redundancy are exploited to design a reactive control strategy driving the robot toward a goal position, properly controlling the interaction between the robot arm and the environment.

In contrast with the current literature (Calandra et al., 2015; Jain et al., 2013; Killpack et al., 2016), that requires to have both tactile sensing capabilities and a torque controlled robot, with the approach presented in the following the robot is controlled at velocity level. Therefore, it can be implemented to robots not equipped with torque sensors at the joints.

8.2 Problem Description

The task considered in this Chapter is to control the motion of the end-effector of a fixed based robot arm to reach a target point, located at a known position $\mathbf{x}_g \in \mathbb{R}^3$, in a cluttered region of the space surrounding the robot: Figure 8.1 shows a test case scenario. The robot does not have any a priori knowledge of the position of the obstacles in the scene and does not make use of visual information and, during the movement toward the target position, it can get in contact with the obstacles with any part of its body.

In order to tackle this problem, the following assumptions have been made. It is assumed that interaction forces can be controlled by using tactile feedback provided by a distributed tactile sensing system integrated all over the robot arm. This system is composed of taxels which are spatially and force calibrated (Kangro et al., 2017), allowing the robot to sense the contact pressure distribution acting on its body and to compute the forces and their locations.

The problem of reconstructing the contact forces acting on an arbitrarily large area of the robot surface using distributed tactile sensors is challenging and computationally complex (Wasko et al., 2019). In the proposed context, for sake of simplicity, it is assumed that the contact area is small and its curvature negligible. Then, considering a connected region of taxels involved into a contact, the equivalent force, applied on the region, is computed as the resultant of the forces sensed by each taxel and applied to the centroid of the area.

Along with the rational proposed in Jain et al. (2013), the following assumptions have been made: (i) slow robot movements have been considered in order to neglect inertial effect in the modelling of the robot dynamic; (ii) friction effects are ignored; (iii) each contact can be described using a linear spring Herzian contact model (Johnson, 1985). Then the effects of a force applied on the contact centroid $\mathbf{x} \in \mathbb{R}^3$ can be modelled as:

$$\dot{\mathbf{f}} = -K\left(\mathbf{n}\mathbf{n}^{\mathsf{T}}\right)\dot{\mathbf{x}} \tag{8.1}$$

where $\dot{\mathbf{x}} \in \mathbb{R}^3$ is the velocity of the contact centroid, $\mathbf{n} \in \mathbb{R}^3$ is the normal to the robot surface at the contact centroid and $K > 0$ is the elastic constant.

8.3 Control Architecture

8.3.1 Tasks definition

The problem of reaching a certain target point $\mathbf{x}_g$ from a generic position $\mathbf{x}_h$ is usually tackled by applying a control law that minimizes a Lyapunov function of the form (Siciliano et al.,

2008):

$$V_g = \frac{1}{2}\mathbf{e}^\intercal \mathbf{e} \tag{8.2}$$

where $\mathbf{e} = \mathbf{x}_h - \mathbf{x}_g$ is the position error.
Suppose the target is not moving and there are no obstacles in contact with the robot arm. The minimization of V_g can be achieved by imposing the following control law (Siciliano et al., 2008):

$$\dot{\mathbf{x}}_h = -\gamma_L \mathbf{e} \tag{8.3}$$

with $\gamma_L > 0$. The Cartesian command can be transformed in joint velocities for the robot with:

$$\dot{\mathbf{q}} = -\gamma_L \mathbf{J}_h^{\#} \mathbf{e} \tag{8.4}$$

where $\mathbf{J}_h$ is the Jacobian matrix associated to the point $\mathbf{x}_h$ and $(\cdot)^{\#}$ is the pseudo-inverse operation.

Now, suppose that at a generic time instant, the robot links are in contact with the environment (see Figure 8.1). In this case, beyond the problem of controlling the motion of the point $\mathbf{x}_h$, the interaction forces arising on the robot arm must also be properly maintained within bounded safety limits. Thus, it is proposed to modify control law in Equation (8.3) by adding the goal of moving the links in contact with the environment, maintaining bounded interaction forces during the robot motion.

$\mathbf{f}_{i,j}$ is then defined representing the j-th force acting on the i-th link, and $\mathbf{n}_{i,j}$ the normal to the robot surface at the contact centroid $\mathbf{x}_{i,j}$.
For each link $i : \{1,2,\ldots,N-1\}$ the minimization of the following function is considered:

$$V_i = \frac{1}{2}\sum_{j=1}^{M_i} f_{i,j}^2 \quad i : \{1,2,\ldots,N-1\} \quad j : \{1,2,\ldots,M_i\} \tag{8.5}$$

where $f_{i,j} = \mathbf{n}_{i,j}^\intercal \mathbf{f}_{i,j} \in \mathbb{R}$ represents the component of the j-th force acting on the i-th link, while N and M_i are respectively the number of robot links and the number of contacts applied on the i-th link at a generic time instant.

For the end-effector, Equation (8.5) is slightly modified by considering the term in Equation (8.2), thus leading to the following expression:

$$V_N = \frac{1}{2}\sum_{j=1}^{M_N} f_{N,j}^2 + \frac{1}{2}\mathbf{e}^\intercal \mathbf{e} \tag{8.6}$$

The minimization of Equations (8.5) corresponds to the minimization of the squared sum of forces acting on the i-th link, while Equation (8.6), defined for the end-effector, beyond the

force terms, also takes into account the motion of $\mathbf{x}_h$ toward the target position.

8.3.2 Exploiting Robot Redundancy to Achieve Multiple Tasks

Equations (8.5) and (8.6) define N tasks (one for each robot link), to be (possibly) executed concurrently, depending on the contact configurations.
Robot redundancy can be exploited to satisfy multiple control objectives. Furthermore, it has already shown to be effective when exploited for problems of obstacle avoidance and collision reaction (Cherubini and Chaumette, 2010; De Luca and Ferrajoli, 2008).
An established control methodology, taking advantage of the manipulator redundancy, is the task priority control architecture (Siciliano and Slotine, 1991). It allows to generate joint level control signals $\dot{\mathbf{q}}$ required to execute a set of tasks with decreasing order of priorities. The method operates so that the execution of a task with high priority can be guaranteed, while the ones with lower priorities will be executed at the best.
For the problem discussed in this Chapter, it is obvious that contacts can arise at any time during the robot motion. For this reason, contact tasks need to be activated or deactivated at run time requiring the use of the priority framework combined with the adoption of activation functions α_i (Lee et al., 2012a), defined in the following of this Section.
The task priority framework applied to the presented case leads to the following structure of nested optimization problems:

$$S_k \triangleq \left\{ \dot{\mathbf{q}} = \underset{\dot{\mathbf{q}} \in S_{k-1}}{\operatorname{argmin}} \left\| \alpha_i \dot{\boldsymbol{\rho}}_i - \alpha_i \mathbf{J}_i \dot{\mathbf{q}} \right\|^2 \right\} \tag{8.7}$$

with $S_0 = \mathbb{R}^N$. In Equation (8.7), k is an index that represents the priority assigned to the corresponding i-th task. S_k is the linear manifold of solutions obtained for the k-th priority level, while α_i, $\dot{\boldsymbol{\rho}}_i$ and J_i are respectively the activation function, the cartesian reference velocity and the Jacobian matrix of the i-th task.
Equation (8.7) states that $\dot{\mathbf{q}}$ must be computed iteratively for each task, starting from the one with higher priority. The solution for the i-th task is found on the residual manifold of solution obtained for previous tasks having higher priority. In the presented case at each time step, the priorities have been assigned starting from the link where the maximum force is applied down to the one with the minimum contact force intensity.

The activation functions α_i are chosen as monolithically increasing functions, taking values in the range [0 1] of the form $\alpha_i = F_{M,i}^4 / F_{thresh}^4$, where $F_{M,i}$ is the maximum force (in terms of absolute value) among those applied to the i-th link and F_{thresh} is the maximum

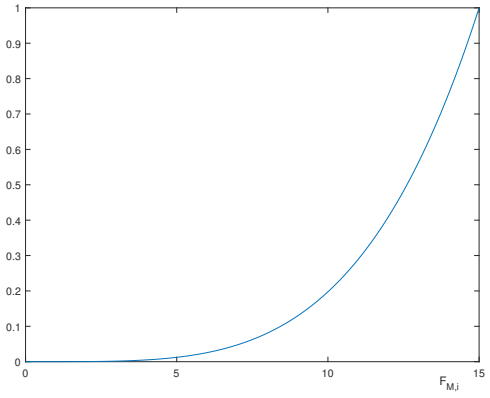


Figure 8.2: Example of the activation function α_i defined in the interval $[0, F_{thesh}]$, with $F_{thresh} = 15$. The value increases depending on the maximum force $F_{M,i}$ applied to the link.

contact force allowed. In this way, α_i smoothly changes depending on the maximum force applied on the i-th link. The activation function is only defined in the interval $[0, F_{thresh}]$ and the shape of α_i for $i : \{1, \ldots, N-1\}$ is shown in Figure 8.2. Since the motion toward the goal must always be enabled, the activation function for the end-effector is $\alpha_N = 1$.

The solution for the problem in Equation (8.7) can be found in Appendix D. In the next Section, it will be described how to compute the cartesian velocity control signal $\dot{\boldsymbol{\rho}}_i$ (see Equation (8.7)), defining the motion of the i-th link, that allows to minimize Equations (8.5) and (8.6). Once the cartesian velocities are defined, the control signal $\dot{\mathbf{q}}$ at joint level can be computed by solving Equation (8.7).

8.4 Control Law Design

8.4.1 Contact Forces Minimization

In order to simplify the notation, the index i is temporarily dropped, since the model is the same for each link.

Given a set of forces acting on the link, to achieve the minimization of Equations (8.5), the control action must impose the following condition at each time instant:

$$\dot{V} = \sum_{j=1}^{M} f_j \dot{f}_j < 0 \tag{8.8}$$

The time derivative of f_j can be computed as follows:

$$\dot{f}_j = \dot{\mathbf{n}}_j^\intercal \mathbf{f}_j + \mathbf{n}_j^\intercal \dot{\mathbf{f}}_j \tag{8.9}$$

From Equation (8.1) it can be seen that $\dot{\mathbf{n}}_j^\intercal \mathbf{f}_j = 0$, since the two vectors are orthogonal. Equation (8.1), representing the contact model, can be substituted in the expression of $\dot{f}_j$, obtaining:

$$\dot{f}_j = \mathbf{n}_j^\intercal \dot{\mathbf{f}}_j = -K_j \mathbf{n}_j^\intercal \left(\mathbf{n}_j \mathbf{n}_j^\intercal\right) \dot{\mathbf{x}}_j = -K_j \mathbf{n}_j^\intercal \dot{\mathbf{x}}_j \tag{8.10}$$

That allows to write $\dot{V}$ as:

$$\dot{V} = -\sum_{j=1}^{M} f_j K_j \mathbf{n}_j^\intercal \dot{\mathbf{x}}_j < 0 \tag{8.11}$$

The velocity of the link at the contact point $\dot{\mathbf{x}}_j$ can be written with respect to a fixed reference frame $< L >$ belonging to the considered link:

$$\dot{\mathbf{x}}_j = \dot{\mathbf{x}}_L + \boldsymbol{\omega}_L \times \mathbf{r}_{j/L} = \dot{\mathbf{x}}_L - \mathbf{r}_{j/L} \times \boldsymbol{\omega}_L \tag{8.12}$$

with $\mathbf{r}_{j/l} = \mathbf{x}_j - \mathbf{x}_L$. Equation (8.12) can be rewritten in a matrix form:

$$\dot{\mathbf{x}}_j = \begin{bmatrix} \mathbf{I} & -\left[\mathbf{r}_{j/L}\times\right] \end{bmatrix} \begin{bmatrix} \dot{\mathbf{x}}_L \\ \boldsymbol{\omega}_l \end{bmatrix} \tag{8.13}$$

where $\mathbf{I}$ is the identity matrix and $\left[\mathbf{r}_{j/L}\times\right]$ is the cross product matrix.
Now, by defining $\mathbf{S}_{j/L} = \left[\mathbf{I} - \left[\mathbf{r}_{j/L}\times\right]\right] \in \mathbb{R}^{3\times 6}$, Equation (8.8) can be written as:

$$\dot{V} = -\sum_{j=1}^{M} f_j K_j \mathbf{n}_j^\intercal \mathbf{S}_{j/L} \begin{bmatrix} \dot{\mathbf{x}}_L \\ \boldsymbol{\omega}_L \end{bmatrix} < 0 \tag{8.14}$$

Finally, with the following definitions:

$$\mathbf{f} = \begin{bmatrix} f_1 & \dots & f_M \end{bmatrix}^\intercal \in \mathbb{R}^M$$

$$\mathbf{H} = \begin{bmatrix} K_1 \mathbf{n}_1^\intercal \mathbf{S}_{1/L} \\ \vdots \\ K_M \mathbf{n}_M^\intercal \mathbf{S}_{M/L} \end{bmatrix} \in \mathbb{R}^{M\times 6} \tag{8.15}$$

$$\dot{\boldsymbol{\rho}} = \begin{bmatrix} \dot{\mathbf{x}}_L & \boldsymbol{\omega}_L \end{bmatrix}^\intercal \in \mathbb{R}^6$$

it is possible to write $\dot{V}$ in a more compact form:

$$\dot{V} = -\mathbf{f}^{\intercal}\mathbf{H}\dot{\boldsymbol{\rho}} < 0 \tag{8.16}$$

The vector $\dot{\boldsymbol{\rho}}$ represents the linear and angular velocity of the current link, which can be intended as the cartesian velocity control action. In order to make $\dot{V}$ negative, $\dot{\boldsymbol{\rho}}$ can be chosen as:

$$\dot{\boldsymbol{\rho}} = \gamma\mathbf{H}^{\#}\mathbf{f} \tag{8.17}$$

where $\mathbf{H}^{\#}$ is the pseudo-inverse (which is properly regularized if it is ill-conditioned) and $\gamma > 0$ is a gain factor.
Equation (8.17) represents a force feedback closed loop control law minimizing the squared sum of the forces applied on $\mathbf{x}_L$.

A control law $\boldsymbol{\rho}_i$, having the form shown in Equation (8.17), is thus defined for each link. These $\boldsymbol{\rho}_i$ are the same appearing in Equation (8.7), which can be transformed in joint velocity commands by solving the corresponding optimization problem.

8.4.2 Motion Toward the Target Position

Now the specific case for the end-effector, related to the minimization of Equation (8.6), is addressed. The control law in Equation (8.17) alone can be used to minimize the first term of V_N, so it will be modified considering the minimization of the second term. Furthermore, it is also assumed that, within the scope of this work, a safe interaction has more importance than reaching the target position. A possible solution to decrease the second term in Equation (8.6), without penalizing the task associated to the contact, is to exploit the null space of $\mathbf{H}_N$. Therefore, $\dot{\boldsymbol{\rho}}_N$ is redefined as:

$$\dot{\boldsymbol{\rho}}_N = \gamma_N\mathbf{H}_N^{\#}\mathbf{f}_N + \mathbf{P}\dot{\mathbf{z}} \tag{8.18}$$

where $\mathbf{P} = \left[\mathbf{I} - \mathbf{H}_N^{\#}\mathbf{H}_N\right]$. Since $\mathbf{P}$ is an orthogonal projector for $\mathbf{H}_N$, the minimization of the first part of V_N will not be affected by the vector $\dot{\mathbf{z}}$, allowing to arbitrarily choose it. Therefore $\dot{\mathbf{z}}$ can be exploited to drive the robot end-effector toward the goal, while it is still in contact with the environment. To this aim, consider Equation (8.3) expressed with respect to a frame $< L >$ rigidly attached to the end-effector:

$$\dot{\mathbf{x}}_h = \mathbf{S}_{h/L}\begin{bmatrix}\dot{\mathbf{x}}_L \\ \boldsymbol{\omega}_L\end{bmatrix} = \mathbf{S}_{h/L}\dot{\boldsymbol{\rho}}_N = -\gamma_L\mathbf{e} \tag{8.19}$$

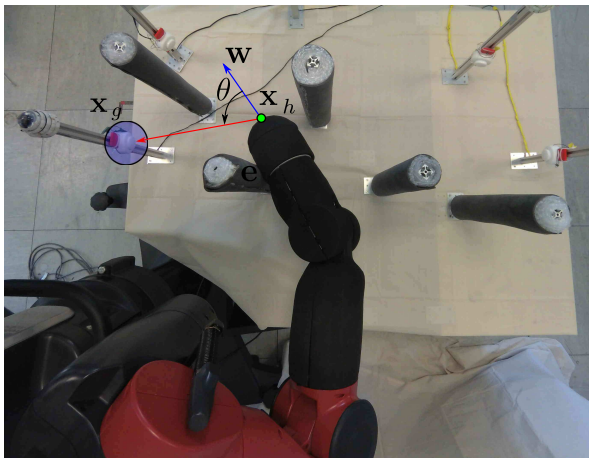


Figure 8.3: The angle between **w** and **e** is minimized, allowing the robot end-effector to rotate around the obstacle.

To substitute Equation (8.18) in Equation (8.19) leads to:

$$\gamma_N \mathbf{S}_{h/L} \mathbf{H}_N^{\#} \mathbf{f}_N + \mathbf{S}_{h/L} \mathbf{P} \dot{\mathbf{z}} = -\gamma_L \mathbf{e} \tag{8.20}$$

Thus the expression for $\dot{\mathbf{z}}$ becomes:

$$\dot{\mathbf{z}} = \left(\mathbf{S}_{h/_L} \mathbf{P}\right)^{\#} \left(-\gamma_L \mathbf{e} - \gamma_N \mathbf{S}_{h/L} \mathbf{H}_N^{\#} \mathbf{f}_N\right) \tag{8.21}$$

As a result, the robot will decrease the contact forces, but at the same time it will move toward the goal if possible. Conversely, when no contacts occur, Equation (8.18) reduces to:

$$\dot{\boldsymbol{\rho}}_N = -\gamma_L \mathbf{S}_{h/L}^{\#} \mathbf{e}. \tag{8.22}$$

8.4.3 Exploiting the End-Effector Orientation

As previously described, the error term **e** is only defined in term of position. However, the orientation of the end-effector is a free parameter that can be exploited to help the robot getting through obstacles.
To this aim, a vector **w** is defined, passing through the centre of the robot end-effector (see Figure 8.3). An idea is to minimize the angle θ between **w** and **e**. In this way, when an obstacle is surpassed, **w** is controlled to point toward $\mathbf{x}_g$, rotating the end-effector around the obstacle.

The minimization of θ can be integrated in the control law $\dot{\boldsymbol{\rho}}_N$. Firstly, there is the need to modify the vector $\mathbf{e}$ in the following way:

$$\mathbf{e} = \begin{bmatrix} \mathbf{x}_g - \mathbf{x}_h \\ [\mathbf{w} \times \mathbf{e}]\theta \end{bmatrix} \in \mathbb{R}^6 \tag{8.23}$$

where $\mathbf{w} \times \mathbf{e}$ defines the rotation axis of the end-effector. Secondly, also the matrix $\mathbf{S}_{h/L}$ must be redefined by considering the angular components of the velocity:

$$\mathbf{S}_{h/L} = \begin{bmatrix} \mathbf{I} & [-\mathbf{r}_{h/L}\times] \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \in \mathbb{R}^{6\times 6} \tag{8.24}$$

Finally, Equation (8.21) is redefined by changing the gain parameter:

$$\dot{\mathbf{z}} = \left(\mathbf{S}_{h/L}\mathbf{P}\right)^{\#} \left(-\Gamma\mathbf{e} - \gamma_N \mathbf{S}_{h/L}\mathbf{H}_N^{\#}\mathbf{f}_N\right) \tag{8.25}$$

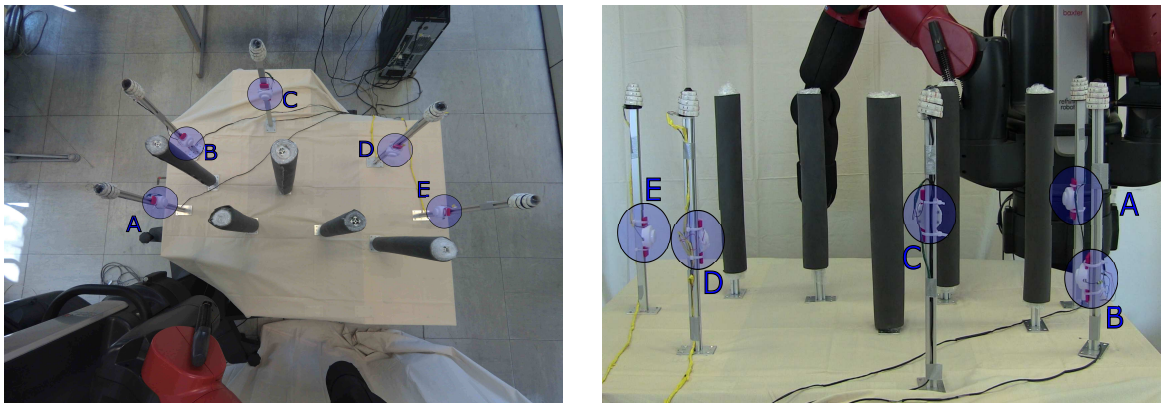
where Γ is a gain matrix defined as

$$\Gamma = \mathrm{diag}(\gamma_L \quad \gamma_L \quad \gamma_L \quad \gamma_A \quad \gamma_A \quad \gamma_A) \tag{8.26}$$

and $\gamma_A > 0$ is the gain term of the orientation task.

8.5 Experiments Description

To validate the proposed approach, the tactile sensing system and the robot presented in Appendixes A and B have been used (see Figure B.3(b)). The controller performance has been validated within the environment shown in Figure 8.4. In the setup there are five target locations, where five switches have been placed. Each of them is connected to a led strip, which will turn on if the switch is pressed.
Five obstacles have been placed between the robot and the targets. The obstacles consist of aluminium extruded bars wrapped in a first layer of pluriball and an external layer of neoprene. The goals and the obstacles have been placed to test the proposed approach in different working conditions. Indeed, some targets are more difficult to reach with respect to others. To have an example consider Figure 8.5, where snapshots of the robot configuration while it is trying to reach a target are shown. It can be seen that depending on the target position one or multiple simultaneous contacts can occur.



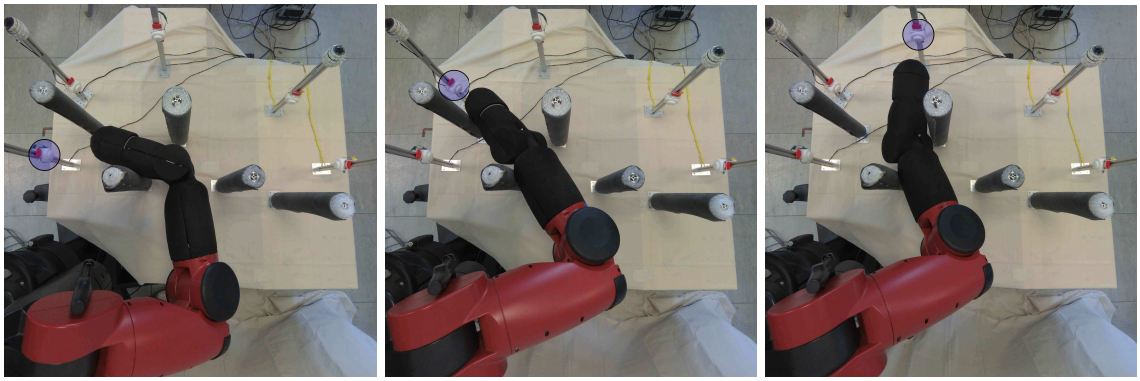
(a) Top view of the experimental setup.

(b) Front view of the experimental setup.

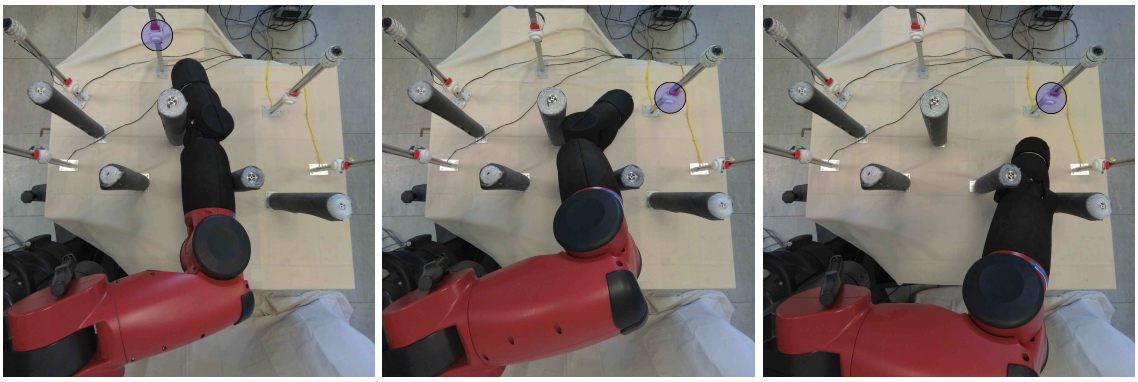
Figure 8.4: Experimental setup. There are 5 obstacles and 5 goal positions.

The proposed method is reactive and so far no planning on top of that has been designed. As previously discussed, the task priority framework used to generate the joint velocity command $\dot{\mathbf{q}}$ cannot guarantee to exactly satisfy all the tasks. Furthermore, the minimization of Equation (8.7) can lead to local minima which mainly depend on the initial arm configuration and the maximum force allowed F_{thresh}. The performances of the proposed method have been evaluated by changing these parameters. In particular, the experiments have been conducted as follows. The robot is commanded to reach each target location 10 times. At the beginning of each experiment the robot arm configuration has been changed. The procedure is repeated twice starting from the same initial conditions, but changing the maximum force threshold. Thus in total 100 experiments have been conducted. We consider the goal successfully reached if an absolute position error of 5 *mm* is achieved without violating the force threshold value F_{thresh}. Conversely, the controller is stopped if the end-effector does not proceed further due to a local minimum or if F_{thresh} is surpassed.
As a remark, it is repeat here that, as previously described in Section 8.3.1 and Section 8.4.2, our controller regulates the squared sum of forces, not the single ones. This means that, depending on the contact configuration, it could happen that the maximum force value acting on a link exceeds the given threshold.

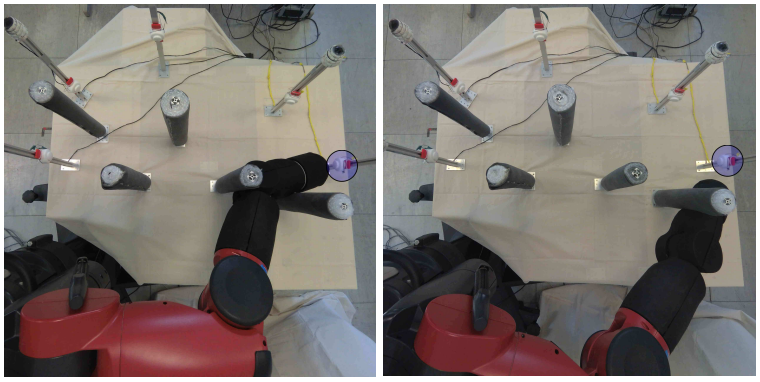
There is only one requirement on the selection of the initial arm configuration. The robot must be able to physically reach the target location. To motivate this assumption consider as an example that the initial position is the one in Figure 8.5(h) and the target location is **C** (see Figure 8.4). Due to the structure of the proposed Lyapunov function, the goal **C** cannot be physically reached starting from that position of the end-effector. This is because the controller will move the arm toward **C** sliding on the obstacle, but the robot arm is not long enough to reach the target position. There is the need of a high level planner that pulls out



(a) (b) (c)



(d) (e) (f)



(g) (h)

Figure 8.5: The robot is moving with the goal of reaching several target positions represented by the blue circles in the images. Depending on the position of the targets there could be very different contact configurations.

the arm and plans a different starting point for the end-effector. This problem is out of the scope of this Chapter and represents an argument for a future extension of the work.

Table 8.1: Summary of the 10 trials executed for each goal location starting from different robot configurations with a force threshold of $15N$.

Goal	**Success Rate**	$\bar{F}$**(N)**	F_m**(N)**	F_M**(N)**	# $F_{max} > F_{thresh}$
A	9 / 10	4.31	2.02	8.98	0
B	10 / 10	3.36	0.72	6.58	0
C	8 / 10	5.26	2.60	8.47	1
D	9 / 10	4.57	1.34	8.33	0
E	7 / 10	4.34	1.63	9.11	0

Table 8.2: Summary of the 10 trials executed for each goal location starting from different robot configurations with a force threshold of $7.5N$.

Goal	**Success Rate**	$\bar{F}$**(N)**	F_m**(N)**	F_M**(N)**	# $F_{max} > F_{thresh}$
A	9 / 10	2.84	1.35	6.52	0
B	9 / 10	2.36	0.71	4.29	1
C	7 / 10	4.86	3.26	6.97	2
D	6 / 10	3.33	1.21	6.83	2
E	6 / 10	3.76	0.81	6.79	3

8.6 Experimental Results

The results are shown in Table 8.1 and Table 8.2, for two different force thresholds. First, the maximum threshold allowed has been set to $15N$. Then it has been halved for the second experiment. The success rate for each target location, along with a summary of the forces applied among the 10 trials, is provided. For each experiment the maximum force applied has been recorded. In Table 8.1 and Table 8.2, $\bar{F}$ is the average of the maximum force applied during each trial, F_m is the smallest maximum force applied and F_M is the maximum force applied over the 10 trials. These statistics are computed for the experiments where the target has been successfully reached. The last column represents how many times the force threshold has been violated. In the other cases the controller stopped due to local minima.

It can be seen that with a F_{thresh} of $15N$ the success rate among all the target is 43/50 which corresponds to 86% and it can be considered a high score for a reactive control approach without planner. Furthermore, the threshold value has been surpassed only once. In the case shown in Table 8.2, where $F_{thresh} = 7.5$, the robot applies lower contact forces, but the overall success rate is reduced to 37/50 which corresponds to 74%.

To better visualize the behaviour of the proposed approach, a video is provided as supplementary material. In the first part, the robot is commanded to reach all the target locations. The second part shows an example of local minimum. The video is split in four

sub-screens. Two of them show the robot motion from the top and the front views. The third sub-screen shows a plot of the maximum force applied for each link, while the last one shows the 3D model of the sensorized robot arm and the number of sensors involved in the contact.

8.7 Conclusion and Discussion

In this Chapter, a low level control law allowing a robot manipulator to get through obstacles has been presented. The interaction forces between the robot and the environment are controlled exploiting distributed tactile sensors placed over the robot arm. No model of the environment or an estimation of it is required. In the experiments the robot has been commanded to reach five target locations starting from different configurations of the arm. Although the actual success rate is at the best 86%, as previously described, there are still limitations related to local minima. The design of a high level planning algorithm can be built on top of the proposed method and it is currently under development. Furthermore, the results shown in Table 8.1 and Table 8.2 suggest a possible extension of the work where the force threshold dynamically changes at run time. Thus the robot will increase the contact forces only if necessary.

Chapter 9

Towards an Architecture for Tactile Based p-HRI

In the previous Chapter, a tactile based control strategy allowing the robot to safely interact with the environment has been proposed. This Chapter focuses on the interaction with humans presenting some preliminary concepts to apply tactile feedback in p-HRI.

As discussed in Section 2.2, tactile sensors have been largely used to translate external forces into robot motions, mostly to ensure a compliant behaviour in response to human inputs. However, more complex situations can be imagined where the robot, instead of being idle, is currently performing a task. The human could actively intervene by physically interacting with the robot, for example, in order to correct or improve its posture while it is still performing the original task. Furthermore, depending on the application, the human could be interested in interacting by moving the whole robot arm, a single joint or to perform more complex actions.

In this sense, the control architecture should be designed to support the execution of multiple concurrent tasks that have to be enabled/disabled or prioritized according to the user input. That said, this scenario suggests that the usage of the task priority framework, already exploited in Chapter 8 to move the robot in unstructured environments, could also fit when interactions with humans are considered. The contents proposed in this Chapter are intended to make a first step in this domain and to introduce possible directions for the future research.

9.1 Introduction and Motivations

Human-robot physical interaction occurs in several ways from simple single contact interactions, e.g. when a human tries to push or pull a robot limb, to complex multiple contacts

conditions where large areas of the robot body are involved, e.g. when a human wants to turn a specific joint or forces a limb into a particular configuration. Moreover, this could happen when the robot is idle or during a task execution, leading to conditions where multiple tasks are required to be satisfied. Therefore, the control architecture should allow to translate contacts into joint motions allowing the execution of multiple concurrent tasks, thus enabling the realization of more complex interactions. Then, since robots are expected to operate in unstructured environments where accidental contacts can occur, they should be able to discriminate between voluntary contacts generated by human physical interaction or generic contacts with humans or objects. In order to implement such kind of behaviour, some key elements are needed: (i) a cognitive level capable of understanding the human intentions; (ii) a multimodal sensory system that captures human input; (iii) a control architecture that executes concurrent tasks with different priorities.

This Chapter preliminarily addresses the topic, by proposing two experiments where the human input is translated in robot motions. The first experiment addresses the possibility of triggering different robot motions depending on the contact configuration. In this context, the task priority architecture and activation functions can be used to enable or disable mutually exclusive tasks. In the second experiment, concurrent tasks are considered. The physical human intervention is managed as an high priority task and activated depending on the semantic of the contact. The results of Chapter 4 are integrated here in order to enable the external input only if a human hand touch is detected.

9.2 Robot Motions Definition

As previously described, one of the proposed applications consists in using the task priority architecture to enable a particular robot motion. The aim is to detect and react accordingly to natural physical interactions that are involved when a human tries to move the robot forearm in the desired configuration. In particular, three different tasks have been defined that can be activated by interacting with the tactile sensors mounted on the Baxter robot (see Appendix B) and described as follows:

- *Task 1*: Move the contact centroid with a linear velocity along the related normal to the surface. This is activated by a single contact on the arm (see Figures 9.1(a) and 9.1(b)) and can be used to move the arm in the desired position.

- *Task 2*: Turn a link. It is activated by grabbing a link with the two hands and by applying a torsion, as in Figure 9.1(c). The tactile sensors are used to detect this

grasping configuration and joint torque sensors are used for understanding the motion direction and intensity.

- *Task 3*: Rotate a joint. It is activated by touching the two links connected to the joint to be moved (see Figure 9.1(d)). Even in this case, the joint torque feedback is exploited to understand the rotation direction and intensity.

9.2.1 Preliminary Definitions

Before formally defining the robot motion tasks to be executed, some preliminary definitions are required and thus presented in the following.
It is assumed that the robot skin provides a set of measurements p_i with $i = \{1, \dots N\}$, where N is the number of tactile elements. If $p_i > 0$ the taxel is considered "active", i.e. a contact is occurring. From these quantities it is possible to define

$$A = \{i\} \text{ iff } p_i > \varepsilon \text{ and } i = \{1, \dots N\}$$

where A is the set with dimension $W \leq N$ which contains the pressure information of the active taxels and ε is a noise threshold. Exploiting A, the *mean pressure* of the active taxels can be defined as:

$$\bar{p} = \frac{\sum_{i \in A} p_i}{W}.$$

Furthermore, assuming a spatially calibrated skin, it is possible to define two sets, $S = \{\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_N\}$ and $K = \{\mathbf{k}_1, \mathbf{k}_2, \dots, \mathbf{k}_N\}$, where $\mathbf{s}_i \in \mathbb{R}^3$ and $\mathbf{k}_i \in \mathbb{R}^3$ are respectively the i-th sensor position and its normal. Using the information provided by the spatial calibration, it is possible to associate at each contact a centroid. Let $\hat{\mathbf{s}} = [\hat{s}_x \ \hat{s}_y \ \hat{s}_z]$ be the center of mass of the active taxel, defined as:

$$\hat{\mathbf{s}} = \frac{\sum_{i \in A} \mathbf{s}_i p_i}{\sum_{i \in A} p_i}$$

Since in a 3D manifold the barycenter of a distribution could not belong to a point on the surface, the contact centroid is considered as the taxel position closest to the center of mass. The centroid $\bar{\mathbf{s}}$ is then defined as:

$$\bar{\mathbf{s}} = \mathbf{s}_h \ : \ h = \underset{i \in A}{\operatorname{argmin}} \ ||\mathbf{s}_i - \hat{\mathbf{s}}|| \tag{9.1}$$

and the associated normal vector is computed in the following equation:

$$\bar{\mathbf{k}} = \frac{\sum_{i \in A} \mathbf{k}_i p_i}{\|\sum_{i \in A} \mathbf{k}_i p_i\|} \tag{9.2}$$

(a) Move the contact centroid (forearm).

(b) Move the contact centroid (wrist).

(c) Turn the arm.

(d) Bend the arm.

Figure 9.1: List of robot motions.

9.2.2 Robot Motion Tasks

In order to enable a particular robot motion, three *mutually exclusive* activation function must be associated to each task. In the specific case, binary activation functions are considered and defined as follows:

$$\alpha_i = \{0,1\} \quad \text{with } i = \{1,2,3\}$$

Each α_i will be triggered by a specific condition:

- a single contact on the sensorized robot arm area will assign $\alpha_1 = 1$, activating the first task (see Figures 9.1(a) and 9.1(b));
- grasping the robot forearm with two hands will impose $\alpha_2 = 1$, allowing to perform a rotation with an angular velocity proportional to the torque applied (see Figure 9.1(c)). This configuration can be detected by evaluating if $\|\sum_{i \in A} \mathbf{k}_i\| < \zeta \approx 0$. This is due to the fact that the sum of the normals of the active taxels should be close to zero during a two-handed embracing grasp;
- two simultaneous contacts performed on the forearm and the wrist will set $\alpha_3 = 1$ (see Figure 9.1(d)).

The different configurations exploited to interact with the robot are summarized in Figure 9.1. Finally, the three task can be formally described in terms of robot velocity reference commands as:

$$\begin{aligned} \dot{\boldsymbol{\rho}}_1 &= -V_{max} \tfrac{\bar{p}-p_{min}}{p_{max}-p_{min}} \left[\bar{k}_1\ \bar{k}_2\ \bar{k}_3\ 0\ 0\ 0\right]^T \\ \dot{\boldsymbol{\rho}}_2 &= -\gamma_2 \left[0\ 0\ 0\ 0\ \tau_f\ 0\ 0\right]^T \\ \dot{\boldsymbol{\rho}}_3 &= -\gamma_3 \left[0\ 0\ 0\ 0\ 0\ \tau_w\ 0\right]^T \end{aligned} \tag{9.3}$$

where τ_f and τ_w are respectively the measured torques applied on the forearm and the wrist. γ_2 and γ_3 are two gain factors. V_{max} represents the maximum speed allowed, while p_{max} and p_{min} are respectively the maximum and minimum full scale pressure values of the sensors. The first vector $\dot{\boldsymbol{\rho}}_1$ represents a reference linear velocity in the Cartesian space, proportional to the pressure applied and directed along the normal of the contact centroid. The other two tasks $\dot{\boldsymbol{\rho}}_2$ and $\dot{\boldsymbol{\rho}}_3$ are defined in the joint space, in order to assign an angular velocity proportional and opposed to the measured torque.

In order to solve the optimization problem (8.7), it is necessary to define three Jacobian matrices. For the first task, $\mathbf{J}_1$ is defined as the Jacobian matrix between the contact centroid and the base frame, projected on the latter. Since the size of $\mathbf{J}_1$ depends on the contact location, if necessary, it will be augmented with zeros to fit the 6×7 size that is the maximum allowed, since the Baxter robot has 7 degrees of freedom. The task Jacobians associated to $\dot{\boldsymbol{\rho}}_2$ and $\dot{\boldsymbol{\rho}}_3$ are J_2 and J_3. These are 7×7 matrices defined as:

$$\begin{aligned} \mathbf{J}_2 &= \text{diag}\{0\,0\,0\,0\,1\,0\,0\} \\ \mathbf{J}_3 &= \text{diag}\{0\,0\,0\,0\,0\,1\,0\} \end{aligned} \tag{9.4}$$

that allow to control a specific joint motion.

The velocity command defined in Equation (9.3) enabled with the corresponding activation function, can be translated in joint velocity command $\dot{\mathbf{q}}$, by solving the problem defined in Equation (8.7)(see Appendix D).

9.3 Experimental Applications

As previously mentioned, two different experimental applications are proposed. In the first one, where it is assumed that the robot is idle, each task defined in the previous Section is activated by a human physical interaction. When one of the three activation functions is enabled the robot will completely follow the human input.

In the second experiment, it is shown how the hierarchical structure of the control architecture can be used for handling high priority human interventions while the robot is performing a task. In this experiment, the human contact recognition system described in Chapter 4 is exploited to recognize the human hand touch.

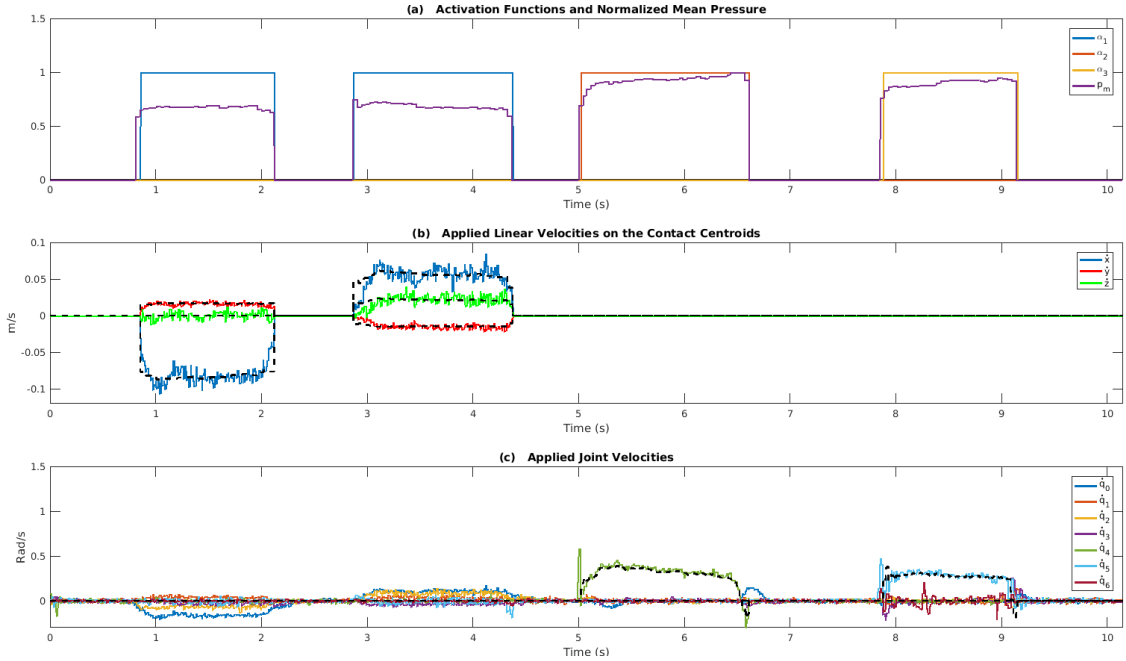


Figure 9.2: Activation of each robot motion task.

9.3.1 Experiment 1: Execution of Robot Motions

The four interactions showed in Figure 9.1 are executed in sequence. Since the motion primitive are mutually exclusive, only one α_i can be non-zero at the same time. Thus, the control law shown in equation (D.11) collapses in the following one (see Appendix D):

$$\dot{\mathbf{q}} = (\alpha_i \mathbf{J}_i)^{\#} \alpha_i \dot{\boldsymbol{\rho}}_i + \left[\mathbf{I} - (\alpha_i \mathbf{J}_i)^{\#} (\alpha_i \mathbf{J}_i)\right] \dot{\mathbf{v}}_i \quad \text{with } \alpha_i = 1 \tag{9.5}$$

Figure 9.2 shows the resulting effects on the robot joint velocities along with the values of the activation functions. In the first two physical interaction, the robot forearm, and successively the wrist, have been pushed away (Figure 9.1(a) 9.1(b)). These interactions have set $\alpha_1 = 1$, producing a linear movement of the contact centroid that follows the reference trajectory indicated by the black dashed line in Figure 9.2 plot (b). Since the contact centroid is defined only when a contact occurs, a zero value of linear velocity is used as a notation in order to indicate the absence of contact. After that, the third interaction involves the forearm only. A torsion is applied to the forearm that starts rotating accordingly, as in Figure 9.1(c). This is reflected in the Figure 9.2 plot (c) where it can be seen that, in the third human-robot interaction, only the joint forearm is moved and $\alpha_2 = 1$. Finally, two contacts occur on both the forearm and the wrist activating the last task, thus setting $\alpha_3 = 1$, that rotates the wrist (as depicted in Figure 9.1(d)).

9.3.2 Experiment 2: High Priority Human Intervention

In this experiment the classification model HandsNet described in Chapter 4 is here used to enable the activation of the task corresponding to $\alpha_1 = 1$. In this case, the model (see Section 4.3) has been implemented using the Tensorlow library (Abadi et al., 2015). The experiment has the goal to emulate a possible operational condition where a robot is performing a straight line path following with the end-effector, at a speed of 0.03 m/s. When a human operator applies a force on the robot arm, the robot should be compliant with respect to the perceived input, but at the same time the path following task should be still executed exploiting the robot redundancy. Furthermore, it has been assumed that if the contact is generated by a human, the robot will accommodate the operator input, otherwise the robot will continue to perform the path following task, ignoring the contact event. To define a strategy to react to non-human contacts is not the focus of this Chapter. A possible control strategy addressing the problem of reacting to collision with the environment or obstacles has already been discussed in Chapter 8.

The timeline of the events is showed in Figure 9.3. During the execution of the low priority task, an accidental contact with the robot arm is performed and the robot is expected to continue executing the path following task. After a while, the human operator starts a cooperation, by pushing the robot forearm in a specific direction. In this case, the contact is classified as a human intervention, activating the task corresponding to $\alpha_1 = 1$ (i.e. the linear movement of the contact centroid), having *high priority*. Figure 9.3, plot (a) shows the value of activation function α_1 and the normalized mean pressure value of the active taxels, in order to clearly show the delay between the human touch and the change of the activation function status. Plots (b) and (c) show the components of the reference and the real linear velocity of the contact centroid expressed with respect to the base reference frame of the robot. Finally, the last plot (d) shows the error of the path following task.
In details, (see Figure 9.3), the robot starts to perform the low priority task. At time $t \cong 2.9$ sec, an accidental contact is generated between the robot forearm and the shoulder of the human operator. A snapshot of the sequence of the contact images is shown in plot (a). The system correctly classifies the contact as generic, without activating the prioritized control law ($\alpha_1 = 0$). Due to the intrinsic compliance of the Baxter robot, the collision increases the task error, as visible in plot (d). When the contact is removed, at $t \cong 3.6$ sec, the robot continues to execute the task.
At time $t \cong 6$ sec, the human operator touches the robot forearm with the hand starting a push operation. A snapshot of the sequence of the contact images is shown above the first plot, where the four fingers are clearly visible. After a delay of about 0.07 sec (which also include tactile data acquisition latency), the Convolutional Neural Network recognizes the

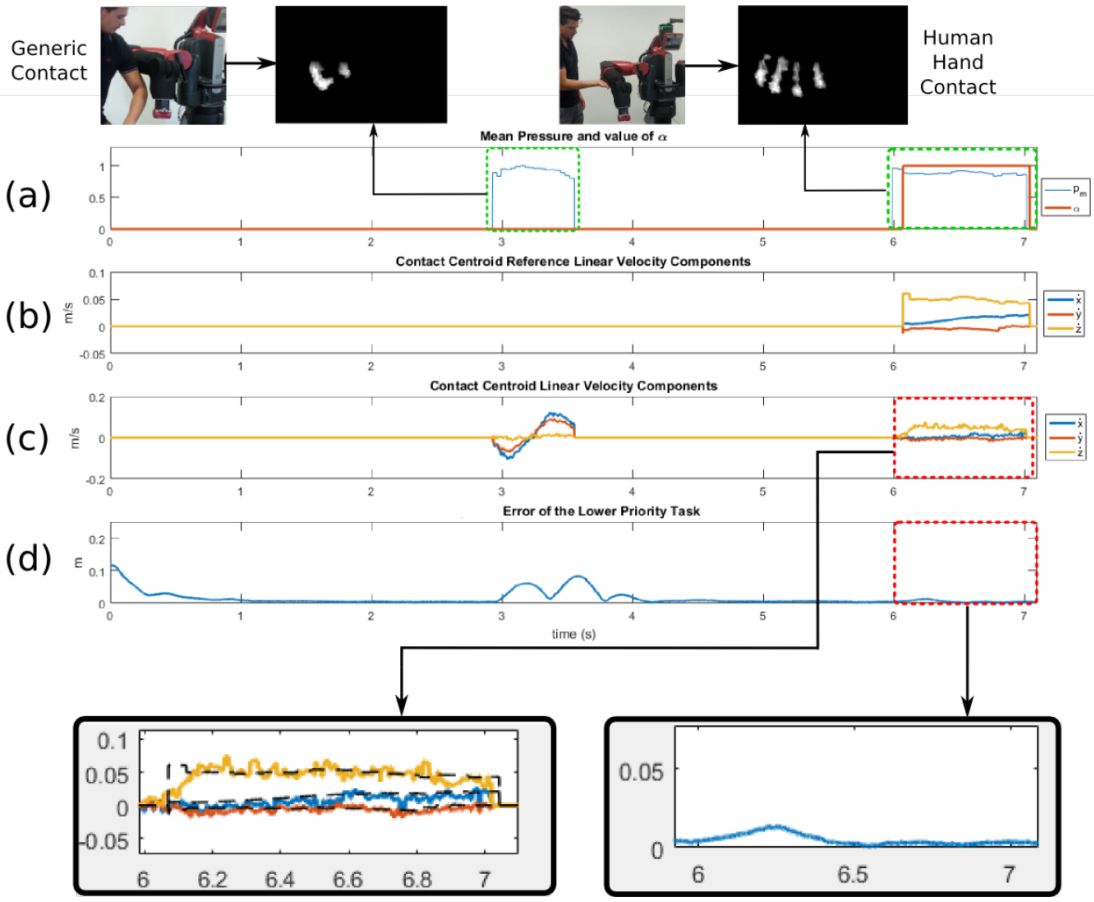


Figure 9.3: Two tasks executed simultaneously. The robot reacts to the contact only if it is generated from a human hand touch.

human touch, triggering the activation function α_1. In turn, the reference velocity signal (second plot), is generated with $V_{Max} = 0.3$ m/s. Again, it must be noted that the velocity of the contact centroid, reported in plot (c), starts to change slightly before that the contact is classified, due to the robot compliance. The zoomed areas of the plots show that the robot moves the contact centroid following the reference value (the black dashed lines represent the reference velocity superimposed on the real velocity achieved). Moreover, exploiting the null space of the high priority task, the robot is able, after a transient, to reach the steady state also in the path following task. It is worth noting that the robot moves the contact centroid following the human input and at the same time, despite the structural compliance of the Baxter, it is still able to perform the path following task, differently from the case of the accidental contact.

9.4 Conclusion and Discussion

In this Chapter two applications exploiting large area tactile sensors in the p-HRI domain have been proposed. What presented here can be useful when approaches based on teach by demonstration are considered. Robots could be taught from scratch when they are in idle state, or online in order to perform small adjustments while the robot is currently performing a task.

Clearly, the proposed contents are not conclusive. The goal of this Chapter is to propose a concept representing one of the possible future directions of this research. In the presented experiments, activation functions are triggered by analysing the contact properties. This aspect can be improved exploiting more sophisticated techniques. As an example, it can be imagined to recognize the human intention or the type of action using machine learning techniques. Indeed, as discussed in Chapter 4, the tactile image generated by a human hand depends on the type of interaction. Thus, the contact shape can be used to discriminate the type of action. Moreover, Section 4.7 shows that the segmentation operation can be used to extract specific information on the contact, such as pressure and areas applied by each finger in contact with the robot body. Other properties could be also extracted by considering also sequences of tactile images. Not to mention that contact information can be integrated with force/torque measurements providing a more complete picture of the contact properties.

Therefore, to provide a method to recognize a the type of interaction enables the realization of a cognitive level on top of the control architecture. Semantic information can be associated to activation functions which in turn trigger specific robot motion task, allowing a human operator to interact with the robot in the most natural way. Although these aspects are not presented in this thesis, they have already been preliminarily addressed and they are currently under development.

Part IV

Conclusions

Chapter 10

Conclusion, Discussion and Future Development

In this last Chapter the contents of the thesis and their contributions are summarized. Then, possible extensions as well as the future research line are discussed.

10.1 Summary and Contribution

This thesis presents novel techniques exploiting large area tactile feedback for contact processing and robot control. The proposed contents can have a major impact in the domain of p-HRI and they are currently applied to an ongoing European project *CoLLaboratE* (grant No 820767), which has the goal of enabling a natural and effective human-robot collaboration in industry scenarios.

It is worth noting that all the presented experiments have been validated on a real robot integrating artificial skin. Contact phenomena over large area are extremely difficult to model and the simulation often does not allow to fully capture their properties. However, robots fully or partially equipped with such a tactile sensing capability are not easily available nowadays. The integration process summarized in Section B has been a fundamental part of the thesis and it took months. It was a necessary step in order to assess the results, by testing the proposed methods on a real hardware rather than on a simulation environment. Moreover, although the experimental setup and the tactile sensor technology were the same among the experiments, the proposed techniques are based on assumptions allowing to be used with other robot skin technologies.

The summary of the work proposed in this thesis is described in the following. Chapter 3 presented a technique to generate tactile images from robot skin measurements. The method

exploits a 2D representation of the robot link allowing to map contacts occurring over the robot body on a image. Tactile images have been applied in Chapter 4, showing that they can be used along with machine learning techniques to discriminate a human hand touch from a generic contact. Furthermore, it has been shown that the pressure distribution can be segmented obtaining more accurate information on the contact distribution. The pipeline proposed in Chapters 3 and 4 represents an improvement of the current state of the art over more than one aspect. Although tactile images and machine learning techniques have already been used in classification tasks (see Sections 2.1 and 2.2), to the best of the author's knowledge, they were applied to small planar tactile sensors distributed onto a regular grid. In the presented case the classification and segmentation tasks have been performed starting from taxels lying over a non-planar manifold with non-regular and non-uniform spatial distribution. Moreover, as discussed in Section 2.2, the possibility of recognizing a human contact using the sense of touch has never been addressed in the literature.

Chapter 5 introduced the problems related to failures and proposed three experiments extending the content of Chapter 4. To the best of the author's knowledge, the problem of assessing the performance of tactile-based classification model in case of varying sensors spatial distribution has never been considered in the state of the art. Chapter 6, extended the concepts presented in Chapter 3, describing the necessary requirements to obtain a whole robot body bi-dimensional representation and proposing a preliminary solution to tackle the problem. The problem of the robot skin spatial calibration mentioned in Section 2.3 has been the subject of Chapter 7, where a novel method to perform an autonomous calibration procedure is proposed. With respect to the current literature, it requires less assumptions extending its applicability to other platform or robot skin technologies. Although the presented procedure requires some key elements that can be added on top of the presented ones, the obtained results prove the validity of the method. Chapter 8 focused on the application of tactile feedback to control the robot motions. In particular, the problem of controlling the interaction between the robot and the environment is addressed. As an operating scenario, the problem of moving the robot arm among obstacles is considered. Although a similar experiment has been proposed in the literature, the novelty lies in the fact that robot skin is used as a direct feedback in the control action. Finally, Chapter 9 presented two experiments introducing the future research objectives. The contents discussed in Chapters 4 and 8 are here considered to integrate cognitive aspects with the robot control.

10.2 Future Directions

The contents proposed in this thesis can be seen as a starting point to develop an architecture exploiting tactile feedback possibly integrating vision and proprioception, allowing robots to better interact with humans or unstructured environments. In the author's view, this architecture can be composed of several *modules*, i.e. software components, that can be interconnected to each other (e.g. human hand touch recognition, actions interpretation, reactive control, etc.).

There are several aspects of the presented work that can be extended. For example, within the scope of the thesis, tactile images have been considered as static objects. Therefore, information related to the dynamic of the contact have not been exploited yet. It can be imagined to integrate information of the contact shape with the dynamic of both time and intensity, thus having the components to develop a system allowing to recognize the type of physical contact. Chapter 4 showed that the possibility of segmenting the contact shape provides information related to applied pressure and areas for each part of the human hand involved in the interaction. Other information, such as the position of the fingers, can be easily extracted. A possibility would be to consider the dynamic of these parameters over time. Although not discussed in the thesis, a preliminary work on sequences of tactile images has already been done and it is currently part of the research activity.

This *cognitive* component can support the control level of the architecture. Although, as proposed in Chapter 8, a purely reactive control law can be imagined, the possibility of recognizing the type of interaction would open more interesting scenarios where the robot can react activating a specific task depending on the semantic of the external input. As an example, suppose that the robot is capable to recognize a gentle contact from a push: in the former case it can behave being compliant to the exerted input, in the latter it can stop for safety reasons.

In aspects related to control, computational problems can arise. Indeed, algorithms providing tactile feedback or data structures (see Chapter 6) must be designed in order to respect hard time constraints. One possible approach would be to have nested control loops, where the innermost one receive raw data at higher frequency. External loops could be slower in order to run algorithms implementing high level behaviours (e.g. the recognition of the type of action) requiring an higher computational effort.

Clearly, the aforementioned techniques must be designed to work on varying size datasets, since depending on the type of contact there could be more or less sensors involved. As it can be guessed, part of the computational problem lies in the quantity of data to be processed. However, new samples do not always bring useful information. Therefore, it can be imagined

to identify meaningful data, studying methods that do not process the whole tactile system, but at the same time do not lose the capabilities of detecting details.

All the aspects discussed above can be extended integrating the sense of touch with vision or joint level measurements. Indeed, beyond the processing of tactile data alone, it is of particular interest to study techniques allowing to fuse information provided by different types feedback, both for enhancing the perception and knowledge on the subject of the current interaction and for compensating the lack of one feedback with another. Cameras provide visual information that could be occluded. Sense of touch can help by capturing the properties of the object (e.g. hardness, size, roughness, etc.) even if not directly visible. Techniques exploiting the integration of vision and touch have already been proposed in the literature to recognize objects (Liu et al., 2017), to localize contacts (Luo et al., 2015a) and to learn visuo-tactile associations (Roncone et al., 2016).

Bibliography

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Arora, R. K. (2015). *Optimization: Algorithms and Applications*, pages 62–66. CRC Press.

Badrinarayanan, V., Kendall, A., and Cipolla, R. (2017). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495.

Berger, A. D. and Khosia, P. K. (1988). Edge Detection For Tactile Sensing. In Sr., W. C. C., editor, *Space Station Automation IV*, volume 1006, pages 163 – 172. International Society for Optics and Photonics, SPIE.

Berger, A. D. and Khosla, P. K. (1991). Using tactile data for real-time feedback. *The International Journal of Robotics Research*, 10(2):88–102.

Beyerer, J., Puente León, F., and Frese, C. (2016). *Morphological Image Processing*, pages 607–647. Springer Berlin Heidelberg, Berlin, Heidelberg.

Bhattacharjee, T., Rehg, J. M., and Kemp, C. C. (2012). Haptic classification and recognition of objects using a tactile sensing forearm. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4090–4097.

Bicchi, A., Salisbury, J. K., and Brock, D. L. (1993). Contact sensing from force measurements. *The International Journal of Robotics Research*, 12(3):249–262.

Billard, A., Calinon, S., Dillmann, R., and Schaal, S. (2008). Robot programming by demonstration. In *Springer handbook of robotics*, pages 1371–1394. Springer.

Brock, O., Trinkle, J., and Ramos, F. (2009). *Super-Flexible Skin Sensors Embedded on the Whole Body, Self-Organizing Based on Haptic Interactions*, pages 294–301. MITP.

Calandra, R., Ivaldi, S., Deisenroth, M. P., and Peters (2015). Learning torque control in presence of contacts using tactile sensing from robot skin. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 690–695.

Cannata, G., Denei, S., and Mastrogiovanni, F. (2010). Towards automated self-calibration of robot skin. In *2010 IEEE International Conference on Robotics and Automation*, pages 4849–4854.

Cannata, G., Maggiali, M., Metta, G., and Sandini, G. (2008). An embedded artificial skin for humanoid robots. In *2008 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pages 434–438.

Cao, L., Kotagiri, R., Sun, F., Li, H., Huang, W., and Aye, Z. M. M. (2016). Efficient spatio-temporal tactile object recognition with randomized tiling convolutional networks in a hierarchical fusion strategy. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 3337–3345. AAAI Press.

Chen, N., Zhang, H., and Rink, R. (1995). Edge tracking using tactile servo. In *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, volume 2, pages 84–89 vol.2.

Cherubini, A. and Chaumette, F. (2010). A redundancy-based approach for obstacle avoidance in mobile robot navigation. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5700–5705.

Cheung, E. and Lumelsky, V. (1989). Development of sensitive skin for a 3d robot arm operating in an uncertain environment. In *Proceedings, 1989 International Conference on Robotics and Automation*, pages 1056–1061 vol.2.

Chitta, S., Jones, E. G., Ciocarlie, M., and Hsiao, K. (2012). Mobile manipulation in unstructured environments: Perception, planning, and execution. *IEEE Robotics Automation Magazine*, 19(2):58–71.

Cirillo, A., Ficuciello, F., Natale, C., Pirozzi, S., and Villani, L. (2016). A conformable force/tactile skin for physical human–robot interaction. *IEEE Robotics and Automation Letters*, 1(1):41–48.

Corke, P. (2013). *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*. Springer Publishing Company, Incorporated, 1st edition.

Cramphorn, L., Ward-Cherrier, B., and Lepora, N. F. (2016). Tactile manipulation with biomimetic active touch. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 123–129.

Dahiya, R. S., Mittendorfer, P., Valle, M., Cheng, G., and Lumelsky, V. J. (2013). Directions toward effective utilization of tactile skin: A review. *IEEE Sensors Journal*, 13(11):4121–4138.

De Luca, A. and Ferrajoli, L. (2008). Exploiting robot redundancy in collision detection and reaction. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3299–3305.

De Luca, A. and Flacco, F. (2012). Integrated control for phri: Collision avoidance, detection, reaction and collaboration. In *2012 4th IEEE RAS EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, pages 288–295.

Del Prete, A., Denei, S., Natale, L., Mastrogiovanni, F., Nori, F., Cannata, G., and Metta, G. (2011). Skin spatial calibration using force/torque measurements. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3694–3700.

Denei, S., Mastrogiovanni, F., and Cannata, G. (2015). Towards the creation of tactile maps for robots and their use in robot contact motion control. *Robotics and Autonomous Systems*, 63:293 – 308. Advances in Tactile Sensing and Touch-based Human Robot Interaction.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.

Desbrun, M., Meyer, M., and Alliez, P. (2002). Intrinsic parameterizations of surface meshes. *Computer Graphics Forum*, 21(3):209–218.

Dhanachandra, N., Manglem, K., and Chanu, Y. J. (2015). Image segmentation using k -means clustering algorithm and subtractive clustering algorithm. *Procedia Computer Science*, 54:764 – 771. Eleventh International Conference on Communication Networks, ICCN 2015, August 21-23, 2015, Bangalore, India.

Duchaine, V. and Gosselin, C. M. (2007). General model of human-robot cooperation using a novel velocity based variable impedance control. In *Second Joint EuroHaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (WHC'07)*, pages 446–451.

Ebert, D. M. and Henrich, D. D. (2002). Safe human-robot-cooperation: image-based collision detection for industrial robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 1826–1831 vol.2.

Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., and Vincent, P. (2010). Why does unsupervised pre-training help deep learning? *Journal of machine learning research : JMLR.*, 11(Feb).

Farfade, S. S., Saberian, M. J., and Li, L.-J. (2015). Multi-view face detection using deep convolutional neural networks. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, pages 643–650. ACM.

Fisher, W. D. and Mujtaba, M. S. (1992). Hybrid position/force control: a correct formulation. *The International journal of robotics research*, 11(4):299–311.

Fortune, S. (1997). Handbook of discrete and computational geometry. chapter Voronoi Diagrams and Delaunay Triangulations, pages 377–388. CRC Press, Inc., Boca Raton, FL, USA.

Friedman, J., Hastie, T., and Tibshirani, R. (1998). Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28:2000.

Frigola, M., Casals, A., and Amat, J. (2006). Human-robot interaction based on a sensitive bumper skin. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 283–287.

Gandarias, J. M., Gómez-de Gabriel, J. M., and García-Cerezo, A. J. (2018). Enhancing perception with tactile object recognition in adaptive grippers for human–robot interaction. *Sensors*, 18(3).

Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., Martinez-Gonzalez, P., and Garcia-Rodriguez, J. (2018). A survey on deep learning techniques for image and video semantic segmentation. *Applied Soft Computing*, 70:41 – 65.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Grau, V., Mewes, A. U. J., Alcaniz, M., Kikinis, R., and Warfield, S. K. (2004). Improved watershed transform for medical image segmentation using prior information. *IEEE Transactions on Medical Imaging*, 23(4):447–458.

Grunwald, G., Schreiber, G., Albu-Schaffer, A., and Hirzinger, G. (2003). Programming by touch: the different way of human-robot interaction. *IEEE Transactions on Industrial Electronics*, 50(4):659–666.

Guo, Y., Liu, Y., Georgiou, T., and Lew, M. S. (2018). A review of semantic segmentation using deep neural networks. *International Journal of Multimedia Information Retrieval*, 7(2):87–93.

Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S., and Lew, M. S. (2016). Deep learning for visual understanding: A review. *Neurocomputing*, 187:27 – 48. Recent Developments on Deep Big Vision.

Haddadin, S., Albu-Schaffer, A., Luca, A. D., and Hirzinger, G. (2008). Collision detection and reaction: A contribution to safe physical human-robot interaction. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3356–3363.

Hoffmann, M., Straka, Z., Farkaš, I., Vavrečka, M., and Metta, G. (2018). Robotic homunculus: Learning of artificial skin representation in a humanoid robot motivated by primary somatosensory cortex. *IEEE Transactions on Cognitive and Developmental Systems*, 10(2):163–176.

Holz, D., Holzer, S., Rusu, R. B., and Behnke, S. (2012). *Real-Time Plane Segmentation Using RGB-D Cameras*, pages 306–317. Springer Berlin Heidelberg, Berlin, Heidelberg.

Horn, R. A. and Johnson, C. R., editors (1985). *Matrix Analysis*. Cambridge University Press, USA.

Hornung, A., Phillips, M., Gil Jones, E., Bennewitz, M., Likhachev, M., and Chitta, S. (2012). Navigation in three-dimensional cluttered environments for mobile manipulation. In *2012 IEEE International Conference on Robotics and Automation*, pages 423–429.

Hoy, M., Matveev, A. S., and Savkin, A. V. (2015). Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey. *Robotica*, 33(3):463–497.

Jain, A., Killpack, M. D., Edsinger, A., and Kemp, C. C. (2013). Reaching in clutter with whole-arm tactile sensing. *The International Journal of Robotics Research*, 32(4):458–482.

Ji, Z., Amirabdollahian, F., Polani, D., and Dautenhahn, K. (2011). Histogram based classification of tactile patterns on periodically distributed skin sensors for a humanoid robot. In *2011 RO-MAN*, pages 433–440.

Johnson, K. L. (1985). *Contact Mechanics*. Cambridge University Press.

Jorda, M., Herrero, E. G., and Khatib, O. (2019). Contact-driven posture behavior for safe and interactive robot operation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9243–9249.

Kaboli, M., Long, A., and Cheng, G. (2015). Humanoids learn touch modalities identification via multi-modal robotic skin and robust tactile descriptors. *Advanced Robotics*, 29(21):1411–1425.

Kangro, J., Traversaro, S., Pucci, D., and Nori, F. (2017). Skin normal force calibration using vacuum bags. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 401–406.

Kato, H. and Harada, T. (2014). Image reconstruction from bag-of-visual-words. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Khatib, O. (1985). Real-time obstacle avoidance for manipulators and mobile robots. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, volume 2, pages 500–505.

Killpack, M. D., Kapusta, A., and C Kemp, C. (2016). Model predictive control for fast reaching in clutter. *Autonomous Robots*, 40:537–560.

Kim, H.-J., Lee, J. S., and Park, J. H. (2008). Dynamic hand gesture recognition using a cnn model with 3d receptive fields. In *2008 International Conference on Neural Networks and Signal Processing*, pages 14–19.

Kimura, H., Horiuchi, T., and Ikeuchi, K. (1999). Task-model based human robot cooperation using vision. In *Proceedings 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human and Environment Friendly Robots with High Intelligence and Emotional Quotients (Cat. No.99CH36289)*, volume 2, pages 701–706 vol.2.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc.

Kuniyoshi, Y., Yorozu, Y., Ohmura, Y., Terada, K., Otani, T., Nagakubo, A., and Yamamoto, T. (2004). *From Humanoid Embodiment to Theory of Mind*, pages 202–218. Springer Berlin Heidelberg, Berlin, Heidelberg.

Leboutet, Q., Dean-León, E., and Cheng, G. (2016). Tactile-based compliance with hierarchical force propagation for omnidirectional mobile manipulators. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pages 926–931.

Lee, J., Mansard, N., and Park, J. (2012a). Intermediate desired value approach for task transition of robots in kinematic control. *IEEE Transactions on Robotics*, 28(6):1260–1277.

Lee, J., Mansard, N., and Park, J. (2012b). Intermediate desired value approach for task transition of robots in kinematic control. *IEEE Transactions on Robotics*, 28(6):1260–1277.

Lepora, N. F., Aquilina, K., and Cramphorn, L. (2017). Exploratory tactile servoing with active touch. *IEEE Robotics and Automation Letters*, 2(2):1156–1163.

Li, Q., Schuermann, C., Haschke, R., and Ritter, H. (2013). A control framework for tactile servoing.

Li, Y. (2012). Hand gesture recognition using kinect. In *2012 IEEE International Conference on Computer Science and Automation Engineering*, pages 196–199.

Liang, H., Yuan, J., and Thalmann, D. (2012). 3d fingertip and palm tracking in depth image sequences. In *Proceedings of the 20th ACM International Conference on Multimedia*, MM '12, pages 785–788, New York, NY, USA. ACM.

Liljebäck, P., Stavdahl, ., Pettersen, K. Y., and Gravdahl, J. T. (2014). Mamba - a waterproof snake robot with tactile sensing. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 294–301.

Lin, H. I., Hsu, M. H., and Chen, W. K. (2014). Human hand gesture recognition using a convolution neural network. In *2014 IEEE International Conference on Automation Science and Engineering (CASE)*, pages 1038–1043.

Liu, H., Greco, J., Song, X., Bimbo, J., Seneviratne, L., and Althoefer, K. (2012). Tactile image based contact shape recognition using neural network. In *2012 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 138–143.

Liu, H., Yu, Y., Sun, F., and Gu, J. (2017). Visual–tactile fusion for object recognition. *IEEE Transactions on Automation Science and Engineering*, 14(2):996–1008.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110.

Luo, S., Mou, W., Althoefer, K., and Liu, H. (2015a). Localizing the object contact through matching tactile features with visual map. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3903–3908.

Luo, S., Mou, W., Althoefer, K., and Liu, H. (2015b). Novel tactile-sift descriptor for object shape recognition. *IEEE Sensors Journal*, 15(9):5001–5009.

Maiolino, P., Maggiali, M., Cannata, G., Metta, G., and Natale, L. (2013). A flexible and robust large scale capacitive tactile system for robots. *IEEE Sensors Journal*, 13(10):3910–3917.

Martinez-Hernandez, U., Metta, G., Dodd, T. J., Prescott, T. J., Natale, L., and Lepora, N. F. (2013). Active contour following to explore object shape with robot touch. In *2013 World Haptics Conference (WHC)*, pages 341–346.

Minato, T., Yoshikawa, Y., Noda, T., Ikemoto, S., Ishiguro, H., and Asada, M. (2007). Cb2: A child robot with biomimetic body for cognitive developmental robotics. In *2007 7th IEEE-RAS International Conference on Humanoid Robots*, pages 557–562.

Mittendorfer, P. and Cheng, G. (2011). Humanoid multimodal tactile-sensing modules. *IEEE Transactions on Robotics*, 27(3):401–410.

Mittendorfer, P., Dean, E., and Cheng, G. (2014). 3d spatial self-organization of a modular artificial skin. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3969–3974.

Mizuuchi, I., Yoshikai, T., Sodeyama, Y., Nakanishi, Y., Miyadera, A., Yamamoto, T., Niemela, T., Hayashi, M., Urata, J., Namiki, Y., Nishino, T., and Inaba, M. (2006). Development of musculoskeletal humanoid kotaro. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 82–87.

Morar, A., Moldoveanu, F., and Gröller, E. (2012). Image segmentation based on active contours without edges. In *2012 IEEE 8th International Conference on Intelligent Computer Communication and Processing*, pages 213–220.

Mukai, T., Onishi, M., Odashima, T., Hirano, S., and Luo, Z. (2008). Development of the tactile sensor system of a human-interactive robot "ri-man". *IEEE Transactions on Robotics*, 24(2):505–512.

Muscari, L., Seminara, L., Mastrogiovanni, F., Valle, M., Capurro, M., and Cannata, G. (2013). Real-time reconstruction of contact shapes for large area robot skin. In *2013 IEEE International Conference on Robotics and Automation*, pages 2360–2366.

Muthukrishnan, C., Smith, D., Myers, D., Rebman, J., and Koivo, A. (1987). Edge detection in tactile images. In *Proceedings. 1987 IEEE International Conference on Robotics and Automation*, volume 4, pages 1500–1505.

Nagi, J., Ducatelle, F., Caro, G. A. D., Cireşan, D., Meier, U., Giusti, A., Nagi, F., Schmidhuber, J., and Gambardella, L. M. (2011). Max-pooling convolutional neural networks for vision-based hand gesture recognition. In *2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*, pages 342–347.

Naya, F., Yamato, J., and Shinozawa, K. (1999). Recognizing human touching behaviors using a haptic interface for a pet-robot. In *IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No.99CH37028)*, volume 2, pages 1030–1034 vol.2.

Nguyen, Bottarel, F., Pattacini, U., Hoffmann, M., Natale, L., and Metta, G. (2018a). Merging physical and social interaction for effective human-robot collaboration. In *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, pages 1–9.

Nguyen, P. D. H., Hoffmann, M., Roncone, A., Pattacini, U., and Metta, G. (2018b). Compact real-time avoidance on a Humanoid Robot for Human-Robot Interaction. *Human-Robot Interaction (HRI), ACM/IEEE International Conference on*.

Ning Chen, Rink, R., and Hong Zhang (1995). Efficient edge detection from tactile data. In *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, volume 3, pages 386–391 vol.3.

Nocedal, J. and Wright, S. J. (2006). *Numerical Optimization*. Springer, New York, NY, USA, second edition.

Ohmura, Y., Kuniyoshi, Y., and Nagakubo, A. (2006). Conformable and scalable tactile sensor skin for curved surfaces. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 1348–1353.

Park, S. and Kwak, N. (2017). Analysis on the dropout effect in convolutional neural networks. In Lai, S.-H., Lepetit, V., Nishino, K., and Sato, Y., editors, *Computer Vision – ACCV 2016*, pages 189–204, Cham. Springer International Publishing.

Pezzementi, Z., Plaku, E., Reyda, C., and Hager, G. D. (2011). Tactile-object recognition from appearance information. *IEEE Transactions on Robotics*, 27(3):473–487.

Pugach, G., Melnyk, A., Tolochko, O., Pitti, A., and Gaussier, P. (2016). Touch-based admittance control of a robotic arm using neural learning of an artificial skin. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3374–3380.

Pugach, G., Pitti, A., and Gaussier, P. (2015). Neural learning of the topographic tactile sensory information of an artificial skin through a self-organizing map. *Advanced Robotics*, 29:1–17.

Raheja, J. L., Chaudhary, A., and Singal, K. (2011). Tracking of fingertips and centers of palm using kinect. In *2011 Third International Conference on Computational Intelligence, Modelling Simulation*, pages 248–252.

Roncone, A., Hoffmann, M., Pattacini, U., Fadiga, L., and Metta, G. (2016). Peripersonal space and margin of safety around the body: Learning visuo-tactile associations in a humanoid robot with artificial skin. *PLOS ONE*, 11(10):1–32.

Roncone, A., Hoffmann, M., Pattacini, U., and Metta, G. (2014). Automatic kinematic chain calibration using artificial skin: Self-touch in the icub humanoid robot. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2305–2312.

Sanfilippo, F., Azpiazu, J., Marafioti, G., Transeth, A. A., Stavdahl, ., and Liljebäck, P. (2016). A review on perception-driven obstacle-aided locomotion for snake robots. In *2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pages 1–7.

Schmidt, P. A., Maël, E., and Würtz, R. P. (2006). A sensor for dynamic tactile information with applications in human–robot interaction and object exploration. *Robotics and Autonomous Systems*, 54(12):1005 – 1014.

Schmitz, A., Maiolino, P., Maggiali, M., Natale, L., Cannata, G., and Metta, G. (2011). Methods and technologies for the implementation of large-scale robot tactile sensors. *IEEE Transactions on Robotics*, 27(3):389–400.

Schmitz, A., Maiolino, P., Maggiali, M., Natale, L., Cannata, G., and Metta, G. (2011). Methods and technologies for the implementation of large-scale robot tactile sensors. *IEEE Transactions on Robotics*, 27(3):389–400.

Schneider, A., Sturm, J., Stachniss, C., Reisert, M., Burkhardt, H., and Burgard, W. (2009). Object identification with tactile sensors using bag-of-features. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 243–248.

Seminara, L., Capurro, M., and Valle, M. (2015). Tactile data processing method for the reconstruction of contact force distributions. *Mechatronics*, 27:28 – 37.

Serino, A. and Haggard, P. (2010). Touch and the body. 34(2):224 – 236. Touch, Temperature, Pain/Itch and Pleasure.

Shimojo, M. (1997). Mechanical filtering effect of elastic cover for tactile sensor. *IEEE Transactions on Robotics and Automation*, 13(1):128–132.

Siciliano, B., Sciavicco, L., Villani, L., and Oriolo, G. (2008). *Robotics: Modelling, Planning and Control*. Springer Publishing Company, Incorporated, 1st edition.

Siciliano, B. and Slotine, J. . E. (1991). A general framework for managing multiple tasks in highly redundant robotic systems. In *Fifth International Conference on Advanced Robotics 'Robots in Unstructured Environments*, pages 1211–1216 vol.2.

Silvera-Tawil, D., Rye, D., and Velonaki, M. (2015). Artificial skin and tactile sensing for socially interactive robots: A review. *Robotics and Autonomous Systems*, 63:230 – 243. Advances in Tactile Sensing and Touch-based Human Robot Interaction.

Simetti, E., Casalino, G., Torelli, S., Sperindé, A., and Turetta, A. (2014). Floating underwater manipulation: Developed control methodology and experimental validation within the trident project. *Journal of Field Robotics*, 31(3):364–385.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556.

Someya, T., Sekitani, T., Iba, S., Kato, Y., Kawaguchi, H., and Sakurai, T. (2004). A large-area, flexible pressure sensor matrix with organic field-effect transistors for artificial skin applications. *Proceedings of the National Academy of Sciences*, 101(27):9966–9970.

Stiehl, W. D. and Breazeal, C. (2005). Affective touch for robotic companions. In Tao, J., Tan, T., and Picard, R. W., editors, *Affective Computing and Intelligent Interaction*, pages 747–754, Berlin, Heidelberg. Springer Berlin Heidelberg.

Stiehl, W. D., Lalla, L., and Breazeal, C. (2004). A "somatic alphabet" approach to "sensitive skin". In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, volume 3, pages 2865–2870 Vol.3.

Sudre, C. H., Li, W., Vercauteren, T., Ourselin, S., and Jorge Cardoso, M. (2017). Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. In Cardoso, M. J., Arbel, T., Carneiro, G., Syeda-Mahmood, T., Tavares, J. M. R., Moradi, M., Bradley, A., Greenspan, H., Papa, J. P., Madabhushi, A., Nascimento, J. C., Cardoso, J. S., Belagiannis, V., and Lu, Z., editors, *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pages 240–248, Cham. Springer International Publishing.

Tawil, D. S., Rye, D., and Velonaki, M. (2011). Improved image reconstruction for an eit-based sensitive skin with multiple internal electrodes. *IEEE Transactions on Robotics*, 27(3):425–435.

Tawil, D. S., Rye, D., and Velonaki, M. (2012). Interpretation of the modality of touch on an artificial arm covered with an eit-based sensitive skin. *The International Journal of Robotics Research*, 31(13):1627–1641.

Tian, S., Ebert, F., Jayaraman, D., Mudigonda, M., Finn, C., Calandra, R., and Levine, S. (2019). Manipulation by feel: Touch-based control with deep predictive models. *CoRR*, abs/1903.04128.

Um, D., Stankovic, B., Giles, K., Hammond, T., and Lumelsky, V. (1998). A modularized sensitive skin for motion planning in uncertain environments. In *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*, volume 1, pages 7–12 vol.1.

Wasko, W., Albini, A., Maiolino, P., Mastrogiovanni, F., and Cannata, G. (2019). Contact modelling and tactile data processing for robot skins. *Sensors*, 19(4).

Wosch, T. and Feiten, W. (2002). Reactive motion control for human-robot tactile interaction. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, volume 4, pages 3807–3812 vol.4.

Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks? In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 27*, pages 3320–3328. Curran Associates, Inc.

Appendix A

The Robot Skin Technology

This Appendix provides an overview on the robot skin technology used in the experiments presented in the manuscript.

A.1 The Cyskin Module

The robot skin used in this work, namely *Cyskin*, is composed of interconnected and conformable modules that allow to cover curved and complex surfaces of the robot body. Cyskin represents an improvement of the technology presented in previous works (Cannata et al., 2008; Schmitz et al., 2011), in terms of:

- possible robot skin shapes and modules arrangement;
- sensor performance;
- measurement synchronization;
- power consumption.

The new robot skin is formed by a mesh of triangular modules allowing to arrange the taxels in a *quasi* regular pattern (each taxel has the same distance from its neighbours, see Figure A.1), with a pitch of about $8\,mm$ (see Figure A.2). The sensors are manufactured using 4-layer flexible printed circuit boards (FPCB) hosting 48 networked modules. Each of them contains 11 capacitive transducers (taxels) and a capacitance to digital converters (CDC) coupled with a microcontroller. With respect to previous prototypes (Cannata et al., 2008; Schmitz et al., 2011), the addition of an on-board microcontroller allows to embed local processing, reconfigurability and a better management of the sensor network.

The taxels are circular pads with a diameter of 3.5 *mm* (Figure A.1), etched on the FPCB and forming one armature of a capacitor. The second armature is formed using a layer of conductive elastic fabric covering the whole robot parts. The capacitive transducers are acquired by AD7147 capacitance to digital converters (manufactured by Analog Devices) providing a resolution of 16 bit.

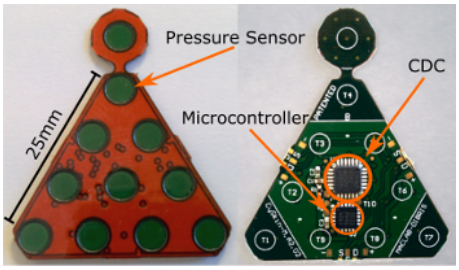


Figure A.1: The Cyskin module.

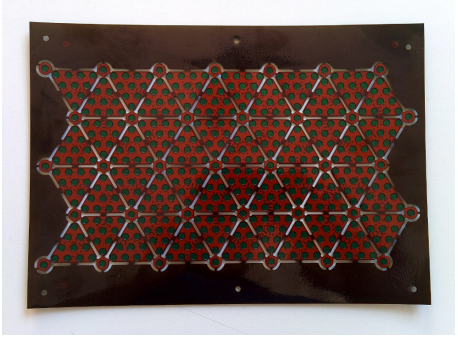


Figure A.2: Cyskin production sheet.

The skin is produced in sheets of 4×12 fully interconnected modules (Figure A.2). The sheet can be cut to the desired shape since all the modules are fully interconnected, while in the previous prototype only a fixed networking pattern was possible. Moreover, the new design maintains the possibility to trim each vertex of each module to further adapt the skin contour to the robot surface to be sensorized.

A.2 The Architecture

The skin architecture consists of two layers (see Figure A.3). The first one is formed by interconnected modules or *skin patches*. The second one is made of Integrated Hub Boards (IHB) (see Figure A.4) that are responsible for measurements synchronization, data aggregation and transmission to the central unit or to the robot controller. The core of an IHB is a reprogrammable microcontroller capable of managing 4 different patches with up to 48 modules each. This increases the number of modules managed by each IHB, with respect to the previous implementation, by three times and dramatically impacts the integration process because less electronics are required with a corresponding reduction of cable harness. The IHBs connect each interconnected module using a single serial bus requiring 4 bus wires and 2 power supply wires for each patch. On the other side, the IHBs communicate with the central unit using a CAN bus.

The microcontroller mounted on each module allows to synchronize the sensor measurements in the whole tactile system. In fact, to start an acquisition, the robot controller sends

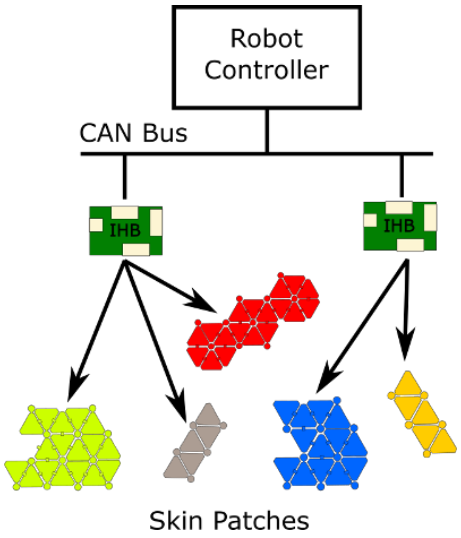


Figure A.3: The robot skin architecture.

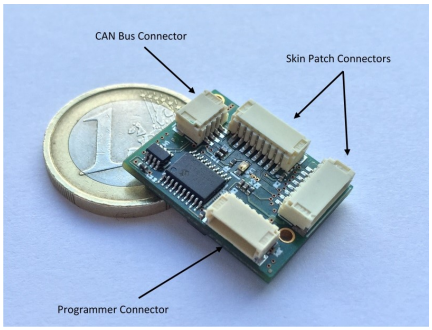


Figure A.4: The integrated hub board.

a broadcast synchronization message to the IHBs that in turn send a broadcast message to all the modules. When the synchronization message is received by a module, it instructs the CDC to start the conversion of all the capacitances. At the end, the IHBs receive the measurements from all the modules and send them back to the PC. In this way, the sensors acquisition is synchronized and a complete snapshot of the tactile system can be periodically acquired. The acquisition cycle is currently set to 50 ms.

Appendix B

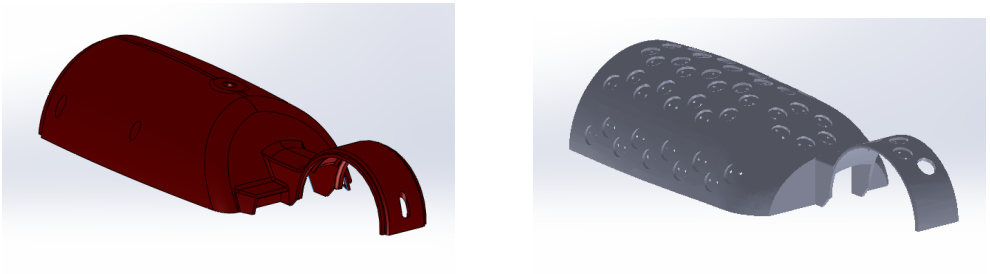
Experimental Platform

This Appendix describes the robot platform used in this thesis and the process performed by the author to integrate the robot skin on an existing robot. Within the scope of this thesis, the integration has been performed on a Baxter robot using the Cyskin technology, but the procedure can be applied to other robots as well.

B.1 Robot Skin Integration Process

The integration process involves several steps detailed in the following:

- The original Baxter covers are redesigned to host the electronics of each skin module. In this phase, the placement of the modules on the robot body is performed manually in order to guarantee an adequate covering of the robot link. Minor changes are applied to the original shape of the covers to allow a better adherence of the conductive fabric covering the sensors. The new 3D model compared to the original one for one cover is shown in Figure B.1. The covers are 3D printed using PLA material.



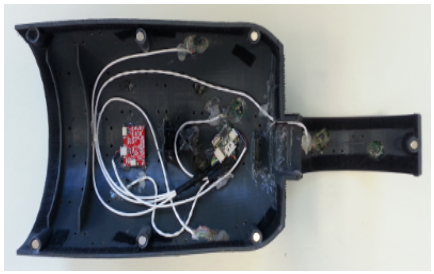
(a) Original Baxter lower forearm cover. (b) New Baxter lower forearm cover.

Figure B.1: Comparison of the original and the custom made cover of the Baxter forearm.

(a) Vacuum machine used for integrating the skin patches.

(b) New Baxter lower forearm cover.

(c) Sensorized robot body part covered by a conductive fabric.



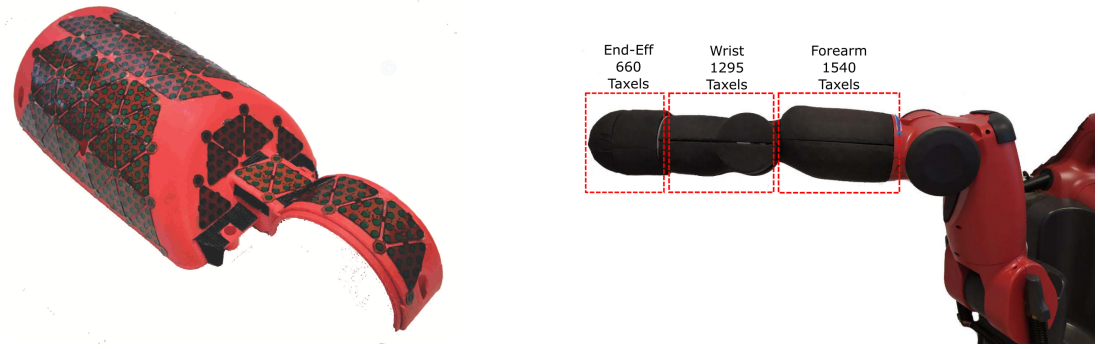
(d) Skin patches are connected inside the cover using an IHB.

Figure B.2: Images describing the robot skin integration process.

- Starting from the production sheet, as in Figure A.2, skin patches are cut following the 3D models design.
- Skin patches are integrated into the covers using the vacuum machine shown in Figure B.2(a). Firstly, some glue is spread on the bottom side of the modules and the patch is positioned on top of the cover. Secondly, the cover is wrapped with a plastic sheet. Finally, the vacuum machine is activated and the wrap is sucked towards the cover surface and kept in place until the glue dries-out. Figure B.2(b) shows the final result of the integration procedure.
- The sensors are covered with a conductive fabric forming a ground plane shielding the sensors, and it is fixed to the covers using velcro strips placed in their inner part. The final result is shown in Figure B.2(c).
- The patches are connected to one or more IHBs that are attached to the robot covers. Due to space constraints, the boards are glued on the surface in the desired position (see Figure B.2(d)).
- Finally, CAN bus cables are routed inside the robot, starting from the robot base, and connected to the IHBs.

B.2 Experimental Platform Details

Two prototypes of Baxter covers have been realized. In the first version, only half of the forearm has been integrated with robot skin. Figure B.3(a) shows the sensorized part, which has been used in the experiments described in Chapters 4 and 5.



(a) First prototype. Half of the robot forearm is integrated with robot skin containing 678 taxels.

(b) Final result: half of the robot arm integrated with tactile sensors and covered with the ground plane.

Figure B.3: The two prototypes created.

In the second version, the forearm, the wrist and the end-effector have been sensorized. This new setup is used in Chapters 6, 8 and 9. The end-effector has also been used in the experiments presented in Chapter 5. The whole setup contains 3495 taxels distributed among the links as detailed in Figure B.3(b) and requires 4 CAN buses to send the data to the robot controller.

Appendix C

Training Details for the Classification and Segmentation Models

This appendix reports information about the training procedure and the hyper-parameters selection for the model discussed in Sections 4.3 and 4.4.

The methodology adopted to find a good set of hyper-parameters is the same among the models. In particular, each model has been subjected to a tuning procedure, where the effects of several possible combinations of hyper-parameters have been investigated. Each combination has been evaluated using *5-fold cross-validation* on the training data (Goodfellow et al., 2016). During the process, a *dataset augmentation* is performed on the training folds: the images have been flipped both horizontally and vertically, increasing the number of images in the training fold by a factor 3.

During the experiments, it was observed that the size of the tactile image has an impact on the model performance. So, we decided to treat it as an hyper-parameter to be tuned.

Images at different resolutions have been tested and at the end it has been kept the one with resolution of 68×100, since it provided the best scores among the models.

As discussed in Section 3, the shape of the tactile images is 247×362. In the case of the **HandsNet**, **BoVW** and **SegNet** we can directly resize and feed images of 68×100 pixels. On the other hand, the **VGG16+SVM**, **VGG16+ft** and **FCN** require an input of 224×224. In order to work with inputs having the same resolution, the tactile images of 68×100 pixels have been padded with zeros in order to fit the shape of 224×224.

C.1 Human Hand Touch Classification

The two networks have been trained in order to minimize the *cross entropy loss* (Goodfellow et al., 2016), using the hyper-parameters reported in Table C.1, where **lr** is the initial learning rate and **lrdf** is a drop factor applied to the learning rate every **lrde** epochs. The other hyper-parameters are the batch size and the number of training epochs.

For what concerns the **VGG+ft** net, the learning rate, defined by the parameters reported in Table C.1, has been applied only in the classification layers. Furthermore, during the training process, the value of the learning rate has been decreased of a 0.1 factor, to fine tune the first three convolutional layers of VGG16.

Table C.1: Hyper-parameters used to train the networks for the classification task.

Model	lr	lrdf	lrde	batch size	epochs
HandsNet	0.01	0.2	40	64	80
VGG+ft	0.1	0.5	40	32	80

In the **VGG+SVM** model, the network works as a feature extractor, so there is no need of training. The classification is performed using a linear SVM, which has been selected by tuning the penalty parameter C. The classifier with $C = 0.25$ has been selected, since it provided the highest accuracy.

In the case of BoVW model, the hyperparameters considered are length of the SIFT descriptors L (Lowe, 2004) and the vocabulary size K (Kato and Harada, 2014), which have been selected as 128 and 80 respectively.

C.2 Human Hand Touch Segmentation

As it can be seen in Figure 4.5, the class distribution is not uniform, indeed most of the pixels (almost 40%) are labelled as *Palm*. A non balanced dataset can cause problems during the training phase since the learning process can be biased in favour of the *Palm* class. As suggested in the literature (Badrinarayanan et al., 2017; Sudre et al., 2017), there are two efficient strategies to deal with an imbalanced dataset. One solution is to use a *cross-entropy* loss weighted using the *median frequency balancing*. Another approach is to use the *dice* loss function. Both methods have been tested. In the case of **SegNet**, the weighted cross-entropy loss performed better, thus it has been selected for training the model. On the contrary, the dice loss produced better results with the **FCN** model.

As described in Long et al. (2015) there are three versions of the **FCN**, namely FCN-32s, FCN-16s and FCN-8s. The difference among them is the size of the stride used in the

Table C.2: Hyper-parameters used to train the networks for the semantic segmentation task.

Model	**lr**	**lrdf**	**lrde**	**batch size**	**epochs**
SegNet	0.1	0.1	90	16	100
FCN	0.1	0.15	80	8	130

classification layer. According to Long et al. (2015) the 8s version provides slightly more accurate predictions. In this work, the FCN-16 has been trained since with the proposed dataset (see Section 4.5) no improvement has been found with respect to the use of FCN-8s, which has a higher computational cost.

The hyper-parameters selected after the tuning procedure are reported in Table C.2. During the training, the models have been fine-tuned by reducing the applied learning rate of a 0.1 factor in the VGG16 convolutional layers, slightly adapting their weights to the new data.

Appendix D

Task Priority Control

This appendix reports the solution to compute the joint control signal $\dot{\mathbf{q}}$ using the task priority approach, generalized considering an arbitrary number of tasks (Lee et al., 2012b; Siciliano and Slotine, 1991; Simetti et al., 2014). The control architecture can be formalized starting from the primary task and recursively by specifying sub-tasks in the null space of the preceding ones. The high priority task can be defined in terms of minimum error norm as:

$$S_1 \triangleq \left\{ \dot{\mathbf{q}} = \underset{\dot{\mathbf{q}}}{\operatorname{argmin}} \left\| \alpha_1 \dot{\boldsymbol{\rho}}_1 - \alpha_1 \mathbf{J}_1 \dot{\mathbf{q}} \right\|^2 \right\} \tag{D.1}$$

where α_1 is a non-negative activation function, $\mathbf{J}_1$ is Jacobian associated to the task, $\dot{\mathbf{q}}$ is the vector of the desired joint velocities and $\dot{\boldsymbol{\rho}}_1$ is the velocity reference.
The solution manifold of the equation (D.1) is:

$$\dot{\mathbf{q}} = (\alpha_1 \mathbf{J}_1)^{\#} \alpha_1 \dot{\boldsymbol{\rho}}_1 + \left[\mathbf{I} - (\alpha_1 J_1)^{\#} (\alpha_1 \mathbf{J}_1) \right] \dot{\mathbf{v}}_1 \quad \forall \, \dot{\mathbf{v}}_1 \tag{D.2}$$

where the first term represents the minimum norm solution, the second corresponds to the null space solution with $\dot{\mathbf{v}}_1$ arbitrarily chosen, and $(\cdot)^{\#}$ is the matrix pseudoinverse operator. From equation (D.2), it can be seen that, if $\alpha_1 = 0$ the cost functional disappear (i.e. the task is not active), generating the following set of solutions

$$\dot{\mathbf{q}} = \dot{\mathbf{v}}_1 \quad \forall \quad \dot{\mathbf{v}}_1 \tag{D.3}$$

Supposing to have another control objective with lower priority, $\dot{\mathbf{v}}_1$ can be exploited to satisfy (if possible) a secondary task, as formalized in Equation (D.4).

$$S_2 \triangleq \left\{ \dot{\mathbf{q}} = \underset{\dot{\mathbf{q}} \in S_1}{\operatorname{argmin}} \left\| \alpha_2 \dot{\boldsymbol{\rho}}_2 - \alpha_2 \mathbf{J}_2 \dot{\mathbf{q}} \right\|^2 \right\} \tag{D.4}$$

This corresponds to solving the following optimization problem:

$$S_2 \triangleq \left\{ \dot{\mathbf{q}} = \dot{\mathbf{r}}_1 + \mathbf{P}_1 \dot{\mathbf{v}}_1 \ : \ \dot{\mathbf{v}}_1 = \underset{\dot{\mathbf{v}}_1}{\operatorname{argmin}} \left\| \alpha_2 \dot{\boldsymbol{\rho}}_2 - \mathbf{H}_2 \dot{\mathbf{r}}_1 - \mathbf{H}_2 \mathbf{P}_1 \dot{\mathbf{v}}_1 \right\|^2 \right\} \tag{D.5}$$

where

$$\begin{array}{rcl} \mathbf{H}_i & = & \alpha_i \mathbf{J}_i, \ i \in \{1,2\} \\ \dot{\mathbf{r}}_1 & = & \mathbf{H}_1^{\#} \alpha_1 \dot{\boldsymbol{\rho}}_1 \\ \mathbf{P}_1 & = & \left[\mathbf{I} - \mathbf{H}_1^{\#} \mathbf{H}_1 \right] \end{array} \tag{D.6}$$

that has the following solution for $\dot{\mathbf{v}}_1$:

$$\begin{array}{r} \dot{\mathbf{v}}_1 = (\mathbf{H}_2 \mathbf{P}_1)^{\#} (\alpha_2 \dot{\boldsymbol{\rho}}_2 - \mathbf{H}_2 \dot{\mathbf{r}}_1) + \\ \left[\mathbf{I} - (\mathbf{H}_2 \mathbf{P}_1)^{\#} (\mathbf{H}_2 \mathbf{P}_1) \right] \dot{\mathbf{v}}_2 \ \ \forall \dot{\mathbf{v}}_2 \end{array} \tag{D.7}$$

The expression above can be substituted in equation (D.2), obtaining the following joint velocities:

$$\begin{array}{r} \dot{\mathbf{q}} = \dot{\mathbf{r}}_1 + \mathbf{P}_1 (\mathbf{H}_2 \mathbf{P}_1)^{\#} (\alpha_2 \dot{\boldsymbol{\rho}}_2 - \mathbf{H}_2 \dot{\mathbf{r}}_1) + \\ \mathbf{P}_1 \left[\mathbf{I} - (\mathbf{H}_2 \mathbf{P}_1)^{\#} (\mathbf{H}_2 \mathbf{P}_1) \right] \dot{\mathbf{v}}_2, \end{array} \tag{D.8}$$

and it can be written in a more compact form by defining $\dot{\mathbf{r}}_2 = \dot{\mathbf{r}}_1 + \mathbf{P}_1 (\mathbf{H}_2 \mathbf{P}_1)^{\#} (\alpha_2 \dot{\boldsymbol{\rho}}_2 - \mathbf{H}_2 \dot{\mathbf{r}}_1)$ and $\mathbf{P}_2 = \mathbf{P}_1 \left[\mathbf{I} - (\mathbf{H}_2 \mathbf{P}_1)^{\#} (\mathbf{H}_2 \mathbf{P}_1) \right]$,

$$\dot{\mathbf{q}} = \dot{\mathbf{r}}_2 + \mathbf{P}_2 \dot{\mathbf{v}}_2 \ \ \forall \dot{\mathbf{v}}_2 \tag{D.9}$$

This process can be iterated on tasks having descending priorities. A recursive solution can be found for $i = \{1, \ldots, L\}$, by defining the following quantities:

$$\begin{array}{c} \mathbf{G}_i = \left[\mathbf{I} - \mathbf{P}_{i-1} (\mathbf{H}_i \mathbf{P}_{i-1})^{\#} \mathbf{H}_i \right] \\ \dot{\mathbf{r}}_i = \mathbf{G}_i \dot{\mathbf{r}}_{i-1} + \mathbf{P}_{i-1} (\mathbf{H}_i \mathbf{P}_{i-1})^{\#} \alpha_i \dot{\boldsymbol{\rho}}_i \\ \mathbf{P}_i = \mathbf{P}_{i-1} \left[\mathbf{I} - (\mathbf{H}_i \mathbf{P}_{i-1})^{\#} (\mathbf{H}_i \mathbf{P}_{i-1}) \right] \end{array} \tag{D.10}$$

which ends with the control law

$$\dot{\mathbf{q}} = \dot{\mathbf{r}}_L + \mathbf{P}_L \dot{\mathbf{v}}_L \ \ \forall \dot{\mathbf{v}}_L. \tag{D.11}$$

Equation (D.11) represents the control law iteratively computed that takes into account the priorities of the tasks and the activation functions α_i allow to enable/disable the tasks. The procedure is initialized with $\dot{\mathbf{r}}_0 = 0$ and $\mathbf{P}_0 = \mathbf{I}$.